Artificial Intelligence to Extract, Analyze and Generate Knowledge and Arguments from Texts to Support Informed Interaction and Decision Making

Elena Cabrio

Université Côte d'Azur

HABILITATION THESIS
Habilitation à Diriger des Recherches (HDR)

Major: Computer Science

Elena CABRIO

Artificial Intelligence to Extract, Analyze and Generate Knowledge and Arguments from Texts to Support Informed Interaction and Decision Making

Jury:
Fabien Gandon, Research Director, INRIA (France) - President
Pietro Baroni, Full Professor, Universita' di Brescia (Italy) - Rapporteur
Marie-Francine Moens, Full Professor, KU Leuven (Belgium) - Rapporteur
Anne Vilnat, Full Professor, Université Paris-Sud (France) - Rapporteur
Chris Reed, Full Professor, University of Dundee (UK) - Examinateur

October 22, 2020

Contents

Acknowledgements

I would like to thank the main authors of the publications that are summarized here for their kind agreement to let these works contribute to this thesis.

Chapter 1

Introduction

This document relates and synthesizes my research and research management experience since I joined the EDELWEISS team led by Olivier Corby in 2011 for a postdoctoral position at Inria Sophia Antipolis (2 years funded by the Inria CORDIS postdoctoral program, followed by 1 year funded by the Labex UCN@SOPHIA and 1 year funded by the SMILK LabCom project). In 2011, the Inria EDELWEISS team and the I3S KEWI team merged to become WIMMICS (Web Instrumented Man-Machine Interactions, Communities and Semantics)[1], a joint team between Inria, University of Nice Sophia Antipolis and CNRS, led by Fabien Gandon. In October 2015, I got an Assistant Professor Position (Maitre de Conference) at the University of Nice Sophia Antipolis, now part of the Université Côte d'Azur. WIMMICS is a sub-group of the SPARKS[2] team (Scalable and Pervasive softwARe and Knowledge Systems) in I3S which has been structured into three themes in 2015. My research activity mainly contributes to the FORUM theme (FORmalising and Reasoning with Users and Models). Throughout this 9-year period, I was involved in several research projects and my role has progressively evolved from junior researcher to scientific leader. I initiated several research projects on my own, and I supervised several PhD thesis. In the meantime, I was also involved in the scientific animation of my research community.

My research area is Natural Language Processing, and the majority of my works are in the sub-areas of Argument(ation) Mining and Information Extraction. Long-term goal of my research (and of works in such research area) is to make computers/machines as intelligent as human beings in understanding and generating language, being thus able: to speak, to make deduction, to ground on common knowledge, to answer, to debate, to support humans in decision making, to explain, to persuade. Natural language understanding can come in many forms. In my research career so far I put efforts in investigating some of these forms, strongly connected to the actions I would like intelligent artificial systems to be able to perform.

In this direction, my first research topic was the study of semantic inferences in natural language texts, that I investigated in the context of my PhD in Information and Communication Technologies (International Doctoral School in Trento, Italy), supervised by Bernardo Magnini at the Fondazione Bruno Kessler, Human Language Technology research group (Trento, Italy). During my PhD, I focused on better understanding the semantic inference needs across NLP applications, and in particular on the Textual Entailment (TE) framework. Textual Entailment has been proposed as a generic framework to model language variability and capture major semantic inference needs across applications in Natural Language Processing. In the TE recognition (RTE) task systems are asked to automatically judge whether the meaning of a portion of text (T), entails the meaning of another text (Hypothesis, H). TE comes at various levels of complexity and involves almost all linguistic phenomena of natural languages.

[1] `https://team.inria.fr/wimmics/`

[2] `https://sparks.i3s.unice.fr/`

Although several approaches have been experimented, TE systems performances are still far from being optimal. My work started from the belief that crucial progress may derive from a focus on decomposing the complexity of the TE task into basic phenomena and on their combination. More specifically, I analyzed how the common intuition of decomposing TE allows a better comprehension of the problem from both a linguistic and a computational viewpoint. I proposed a framework for component-based TE, where each component is in itself a complete TE system, able to address a TE task on a specific phenomenon in isolation. I investigated the following dimensions: i) the definition and implementation of a component-based TE architecture; ii) the linguistic analysis of the phenomena; iii) the automatic acquisition of knowledge to support entailment judgments; iv) the development of evaluation methodologies to assess TE systems capabilities. During my PhD, I have also spent 6 months at the Xerox Research Center Europe in Grenoble (now NAVER Labs Europe), in the Parsing and Semantics research lab, where I continued my research on my PhD topic.

Following the long-term goal mentioned above and after defending my PhD, my research topics gradually evolved from the study of semantic inferences between textual snippets to the investigation of methods to extract structured information from unstructured natural language text, to populate knowledge bases in different application scenarios, with the goal of making intelligent system ground on common knowledge. To enhance users interactions with such structured data, I then addressed the challenge of mapping natural language expressions (e.g., user queries) with concepts and relations in structured knowledge bases, implementing a question answering system. The following step was to focus on mining and analyzing argumentative structures (from heterogeneous sources, but mainly from the Web), as well as connecting them with datasets on the Web of Data for their interlinking and semantic enrichment. I am one of the very first initiators of the research topic, very popular nowadays, called Argument Mining. The rationale behind my research work (as one of the core topics of the WIMMICS team) is to support structured argument exchange, informed decision making and improved fact-checking, also considering the role of emotions and persuasion. For this reason, argumentative structures and relevant factual information should be mined, understood and interlinked, and the Web represents both an invaluable information source, and the ideal place to publish the mined results, producing new types of links in the global vision of weaving a richer and denser Web.

To preserve an open, safe and accessible Web, another aspect of increasing importance in this scenario is the development of robust systems to detect abuse in online user generated content. In this context, we addressed the task of multilingual abusive language detection, taking advantage of both the network analysis and the content of the short-text messages on online platforms to detect cyberbullying phenomena.

This chapter is organized as follows: Sections 1.1, 1.2, 1.3 and 1.4 provide an overview of the four main research areas my research contributions deal with. More specifically, in each section, I provide: a survey of the research area and a description of the projects in which I am or was involved, presented by application domains. The scientific contributions themselves are detailed in the next chapters.

1.1 Information Extraction to generate structured knowledge

Information extraction (IE) is the process of extracting specific (pre-specified) information from textual sources. Gathering detailed structured data from texts (where there is a regular and predictable organization of entities and relationships), information extraction enables the automation of tasks such as smart content classification, integrated search, management and delivery, and data-driven activities such as mining for patterns and trends, uncovering hidden

relationships, and so on. Broadly speaking, the goal of IE is to allow computation to be done on the previously unstructured data. Typically, for structured information to be extracted from unstructured texts, the following main subtasks are involved: (i) pre-processing of the text, (ii) finding and classifying concepts (e.g., mentions of people, things, locations, events and other pre-specified types of concepts detected and classified), (iii) connecting the concepts (i.e. identifying relationships between the extracted concepts), (iv) eliminating duplicate data, and (v) enriching a knowledge base for further use.

Natural language remains the most natural means of interaction and externalization of information for humans. Manually managing and effectively making use of the growing amount of information available in unstructured form (e.g., news, articles, social media) is tedious, boring and labor intensive. Information Extraction systems take natural language text as input and produce structured information specified by certain criteria, that is relevant to a particular application.

I have been conceived and applied Information Extraction methods in different application scenarios, and in the context of several projects.

I was the French project leader of the ALOOF project "Autonomous learning of the meaning of objects" (funded by CHIST-ERA[3], 2015-2018), whose goal was to enable robots and autonomous systems working with and for humans to exploit the vast amount of knowledge on the Web in order to learn about previously unseen objects involved in human activities, and to use this knowledge when acting in the real world. In particular, the goal of our research was to mine relevant knowledge from the Web on domestic settings (including objects' names; class properties; appearance and shape; storage locations; and typical usages/functions). We presented a framework for extracting such knowledge in the form of (binary) relations. It relied on a ranking measure that, given an object, ranks all entities that potentially stand in the relation in question to the given object. More precisely, we relied on a representational approach that exploits distributional spaces to embed entities into low-dimensional spaces in which the ranking measure can be evaluated. In the context of this project I supervised the work of Valerio Basile and Roque Lopez Condori, research engineers.

I was also involved in the WASABI project, funded by ANR[4], 2016-2019, whose goal is the creation of a 2 million song knowledge base that combines metadata collected from music databases on the Web, metadata resulting from the analysis of song lyrics, and metadata resulting from the audio analysis, and the development of semantic applications with high added value to exploit this semantic database. In the context of such project, I was co-supervising the Ph.D. thesis of Michael Fell on Natural Language Processing of song lyrics. Given that lyrics encode an important part of the semantics of a song, our research activity in this context focused on extracting relevant information from the lyrics, such as their structure segmentation, their topics, the explicitness of the lyrics content, the salient passages of a song and the emotions conveyed. Together with researchers at IRCAM (Paris), partners of the WASABI project, we have investigated the coupling of text and audio to improve system performances for Music Information Retrieval.

The purpose of the LabCom SMILK "Social Media Intelligence and Linked Knowledge" (common laboratory between WIMMICS and the company ViseoGroup in Grenoble[5], 2014-2017) was to develop research and technologies in order to retrieve, analyze, and reason on textual data coming from Web sources, and to make use of Linked Open Data, social networks structures and interaction in order to improve the analysis and understanding of textual resources. In the context of this project I supervised the work of Farhad Nooralahzadeh, research engineer (now PhD at the University of Oslo), on semantically enriching raw data by linking

[3] https://project.inria.fr/aloof/2015/09/22/the-aloof-project/

[4] http://wasabihome.i3s.unice.fr/

[5] project.inria.fr/smilk/

the mentions of named entities in the text to the corresponding known entities in knowledge bases. In the context of the same project, we experimented brand-related information retrieval in the field of cosmetics. We created the ProVoc ontology to describe products and brands, we automatically populated a knowledge base mainly based on ProVoc from heterogeneous textual resources, and we developed a browser plugin providing additional knowledge to users browsing the web relying on such ontology.

In the Topic Detection and Tracking (TDT) community, the task of analyzing textual documents for detecting events is called Event Detection (ED). ED is generally defined as a discovery problem, i.e., mining a set of documents for new patterns recognition [487]. It mainly consists in discovering new or tracking previously identified events. ED from texts were mainly focused on finding and following events using conventional media sources such as a stream of broadcast news stories. In the latest years, NLP researchers have shown growing interest in mining knowledge from social media data, specially Twitter, to detect and extract structured representations and summarize newsworthy events. In this context, I have supervised the PhD thesis of Amosse Edouard on studying methods for detecting, classifying and tracking events on Twitter.

1.2 Natural language interaction with the Web of Data

While more and more structured data is published on the web, the question of how typical web users can access this body of knowledge keeps on being of crucial importance. Over the past years, there has been a growing amount of research on interaction paradigms that allow end users to profit from the expressive power of Semantic Web standards while at the same time hiding their complexity behind an intuitive and easy-to-use interface. Especially natural language interfaces have received wide attention, as they allow users to express arbitrarily complex information needs in an intuitive fashion and, at least in principle, in their own language. Multilingualism is, in fact, an issue of major interest for the Semantic Web community, as both the number of actors creating and publishing data all in languages other than English, as well as the amount of users that access this data and speak native languages other than English is growing substantially.

The key challenge is to translate the users' information needs into a form such that they can be evaluated using standard Semantic Web query processing and inferencing techniques. Over the past years, a range of approaches have been developed to address this challenge, showing significant advances towards answering natural language questions with respect to large, heterogeneous sets of structured data [449]. However, only few systems yet address the fact that the structured data available nowadays is distributed among a large collection of interconnected datasets, and that answers to questions can often only be provided if information from several sources are combined. In addition, a lot of information is still available only in textual form, both on the web and in the form of labels and abstracts in linked data sources. Therefore approaches are needed that can not only deal with the specific character of structured data but also with finding information in several sources, processing both structured and unstructured information, and combining such gathered information into one answer [402, 451].

In this context, I have addressed the problem of enhancing interactions between non-expert users and data available on the Web of data, focusing in particular on the automatic extraction of structured data from unstructured documents to populate RDF triple stores, and in a Question Answering (QA) setting, on the mapping of natural language expressions (e.g., user queries) with concepts and relations in a structured knowledge base. In particular, I have designed and implemented the models and algorithms of a Question Answering system over Linked Data (QAKiS) based on DBpedia as the RDF data set to be queried using a natural language interface. To reconcile information obtained by language specific DBpedia chapters, I have integrated an argumentation-based module into the system to reason over inconsistent

information sets, so as to provide a unique and justified answer to the user.

1.3 Mining argumentative structures from texts

In the last years, the growing of the Web and the daily increasing number of textual data published there with different purposes have highlighted the need to process such data in order to identify, structure and summarize this huge amount of information. Online newspapers, blogs, online debate platforms and social networks, but also normative and technical documents provide an heterogeneous flow of information where natural language arguments can be identified, and analyzed. The availability of such data, together with the advances in Natural Language Processing and Machine Learning, supported the rise of a new research area called Argument(ation) Mining (AM). AM is a young and emerging research area within computational linguistics. At its heart, AM involves the automatic identification of argumentative structures in free text, such as the conclusions, premises, and inference schemes of arguments as well as their interrelations and counter-considerations [92].

Two main stages have to be considered in the typical argument mining pipeline, from the unstructured natural language documents towards structured (possibly machine-readable) data:

Arguments' extraction: The first stage of the pipeline is to detect arguments within the input natural language texts. Referring to standard argument graphs, the retrieved arguments will thus represent the nodes in the final argument graph returned by the system. This step may be further split in two different stages such as the extraction of arguments and the further detection of their boundaries.

Relations' extraction: The second stage of the pipeline consists in constructing the argument graph to be returned as output of the system. The goal is to predict what are the relations holding between the arguments identified in the first stage. This is an extremely complex task, as it involves high-level knowledge representation and reasoning issues. The relations between the arguments may be of heterogeneous nature, like attack, support or entailment [24]. This stage is also in charge of predicting, in structured argumentation, the internal relations of the argument's components, such as the connection between the premises and the claim [11, 77]. Being it an extremely challenging task, existing approaches assume simplifying hypotheses, like the fact that evidence is always associated with a claim [1].

To date, researchers have investigated AM on genres such as legal documents [330, 18], product reviews [456, 481], news articles [178, 5], online debates [89, 461], user-generated web discourse [364, 204, 154], Wikipedia articles [274, 396], academic literature [265], bioscience literature [202], persuasive essays [425], and dialogues [74, 6]. Recently, also argument quality assessment came into focus [205, 162]. AM is also inherently tied to stance and sentiment analysis [25], since every argument carries a stance towards its topic, often expressed with sentiment. Argument mining gives rise to various practical applications of great importance. In particular, it provides methods that can find and visualize the main pro and con arguments in a text corpus – or even in an argument search on the web – towards a topic or query of interest [396, 423]. In instructional contexts, written and diagrammed arguments represent educational data that can be mined for conveying and assessing students' command of course material [352].

In this context, together with Serena Villata (CNRS), I proposed a combined framework of natural language processing and argumentation theory to support human users in their interactions. The framework combines a natural language processing module which exploits the Textual Entailment (TE) approach (previously investigated during my PhD thesis) and detects the arguments in natural language debates and the relationships among them, and an argu-

mentation module which represents the debates as graphs and detects the accepted arguments. The argumentation module is grounded on bipolar argumentation. Moreover, I studied the relation among the notion of support in bipolar argumentation, and the notion of TE in Natural Language Processing (NLP). This research line about natural models of argumentation resulted in the NoDE Benchmark[6], a benchmark of natural arguments extracted from different kinds of textual sources. It is composed of three datasets of natural language arguments, released in two machine-readable formats, i.e., the standard XML format, and XML/RDF format adopting the SIOC-Argumentation vocabulary (extended). Arguments are connected by two kinds of relations: a positive (i.e., support) relation, and a negative (i.e., attack) relation, leading to the definition of bipolar argumentation graphs. I started also to investigate the mapping between argumentation schemes in argumentation theory, and discourse in Natural Language Processing, together with Serena Villata (CNRS) and Sara Tonelli (FBK Trento).

This research line continued by applying argument mining to political speeches. I also co-supervise (official supervisors are Serena Villata and Leendert van der Torre), the PhD thesis of Shohreh Haddadan (University of Luxembourg – 2017-2020) about "Argument Mining on Political debates". Given that political debates offer a rare opportunity for citizens to compare the candidates' positions on the most controversial topics of the campaign, we carried out an empirical investigation of the typology of argument components in political debates by annotating a set of political debates from the last 50 years of US presidential campaigns, creating a new corpus of 29k argument components, labeled as premises and claims.

In the same line, I co-supervise also (official supervisors are Serena Villata and Celine Poudat), the PhD thesis of Tobias Mayer on "Argument Mining on medical data" (in particular, so far we focused on extracting argumentative structures and their relations from Randomized Clinical Trials). Evidence-based decision making in the health-care domain targets at supporting clinicians in their deliberation process to establish the best course of action for the case under evaluation. We proposed a complete argument mining pipeline for RCTs, classifying argument components as *evidence* and *claims*, and predicting the relation, i.e., *attack* or *support*, holding between those argument components on a newly created dataset of RCTs. We experiment with deep bidirectional transformers in combination with different neural architectures and outperformed current state-of-the-art end-to-end argument mining systems. These set of works are carried out in the context of the IADB project (Intégration et Analyse de Données Biomédicales, 2017-2020) funded by IDEX UCA Jedi[7]. The goal of the project is to define the methods to access, process and extract information from a huge quantity of biomedical data from heterogeneous source and of different nature (e.g., texts, images).

Another scenario to which I have contributed with argument mining methods concerned the extraction of argumentative text from social media short messages in the context of the Inria CARNOT Project "Natural Language Argumentation on Twitter" (2014-2015) with the start-up Vigiglobe, based in Sophia Antipolis. Understanding and interpreting the flow of messages exchanged in real time on social platforms, like Twitter, raises several important issues. The big amount of information exchanged on these platforms represents a significant value for who is able to read and enrich this multitude of information. Users directly provide this information, and it is interesting to analyze such data both from the quantitative and from the qualitative point of view, especially for what concerns reputation and marketing issues (regarding brands, institutions or public actors). Moreover, the automated treatment of this type of data and the constraints it presents (e.g., limited number of characters, tweets with a particular writing style, amount of data, real-time communication) offer a new and rich context for a challenging use of existing tools for argument mining. In the context of this project, I supervised the activity of Tom Bosc, a research engineer, now doing a PhD in Canada. Always in the context of this

[6] http://www-sop.inria.fr/NoDE/

[7] http://www.i3s.unice.fr/~riveill/IADB/

project, I supervised the 3-month internship of Mihai Dusmanu (Ecole Normale Superieure, Paris) about Argument Detection on Twitter.

Still on argument mining, I am currently involved in the DGA RAPID CONFIRMA project "COntre argumeNtation contre les Fausses InfoRMAtions". Counter-argumentation is a process aiming to put forward counter-arguments in order to provide evidences against a certain argument previously proposed. In the case of fake news, in order to convince a person that the (fake) information is true, the author of the fake news uses different methods of persuasion via arguments. Thus, goal of our research is to propose performing methods to identifying these arguments and attacking them by proposing carefully constructed arguments from safe sources, as a way to fight this phenomenon and its spread along the social network. In the context of this project, I am co-supervising the postdoc of Jerome Delobelle.

My research activity dealing with emotions in argumentation started with the SEEMPAD "Social Exchanges and Emotions in Mediated Polemics - Analysis and Data Associate"[8] project (Joint Research Team with Heron Lab, 2014-2016). The goal of this project was to study the different dimensions of exchanges arising on online discussion platforms. More precisely, we concentrated on the study and analysis of the impact of emotions and mental states on the argumentation in online debates, with a particular attention to the application of argumentative persuasion strategies.

Recently, the ANSWER Inria-Qwant research project[9] was launched, whose goal is to develop the new version of the Qwant search engine by introducing radical innovations in terms of search criteria as well as indexed content and users' privacy. In this context, I am co-supervising the PhD thesis of Vorakit Vorakitphan (2018-2021) on extracting effective and scalable indicators of sentiment, emotions, and argumentative relations in order to offer the users additional means to filter the results selected by the search engine.

1.4 Cyberbullying and abusive language detection

The use of social media platforms such as Twitter, Facebook and Instagram has enormously increased the number of online social interactions, connecting billions of users, favouring the exchange of opinions and giving visibility to ideas that would otherwise be ignored by traditional media. However, this has led also to an increase of attacks targeting specific groups of users based on their religion, ethnicity or social status, and individuals often struggle to deal with the consequences of such offenses.

This problem affects not only the victims of online abuse, but also stakeholders such as governments and social media platforms. For example, Facebook, Twitter, YouTube and Microsoft have recently signed a code of conduct[10], proposed by the European Union, pledging to review the majority of valid notifications for removal of illegal hate speech in less than 24 hours.

Within the Natural Language Processing (NLP) community, there have been several efforts to deal with the problem of online abusive detection, since the computational analysis of language can be used to quickly identify offenses and ease the removal of abusive messages. Several workshops [471, 182] and evaluation campaigns [179, 70, 477, 38, 493] have been recently organised to discuss existing approaches to hate speech detection, propose shared tasks and foster the development of benchmarks for system evaluation. These have led to the creation of a number of datasets for hate speech detection in different languages, that have been shared within the NLP research community. Recent advances in deep learning approaches to text classification have then been applied also to deal with this task, achieving for some languages state-of-the-art

[8] https://project.inria.fr/seempad/

[9] `https://project.inria.fr/answer/`

[10] `http://ec.europa.eu/justice/fundamental-rights/files/hate_speech_code_of_conduct_en.pdf`

results [116, 185, 190]. These systems are usually tailored to deal with social media texts by applying pre-processing, using domain-specific embeddings, adding textual features, etc.

Related to this context, I am the French leader of the EIT CREEP (Cyberbulling Effect Prevention, 2017-2019) project (funded by EIT Digital[11]). The purpose of CREEP is to provide a set of tools to support the detection and prevention of psychological/behavioral problems of cyberbullying teenage victims. The objective is achieved combining social media monitoring and motivational technologies (virtual coaches integrating chatbots). From February 2018 to October 2019, I have supervised the activity of two research engineers, Pinar Arslan and Michele Corazza, working on a multilingual platform to monitor cyberbullying based on message classification and social network analysis.

Together with them and Sara Tonelli (FBK, Trento, partner of the CREEP project) we proposed a deep learning architecture for abusive detection that is rather stable and well-performing across different languages to evaluate the endowments of several components that are usually employed in the task, namely the type of embeddings, the use of additional features (text-based or emotion-based), the role of hashtag normalisation and that of emojis. We performed our comparative evaluation on English, Italian and German, focusing on freely available Twitter datasets for hate speech detection. Our goal is to identify a set of recommendations to develop hate speech detection systems, possibly going beyond language-specific differences.

Moreover, we performed a comparative evaluation on freely available datasets for hate speech detection in Italian, extracted from four different social media platform, i.e. Facebook, Twitter, Instagram and Whatsapp, to understand if it would it be advisable to combine such platform-dependent datasets to take advantage of training data developed for other platforms in low-resource scenarios. We have integrated the proposed classifiers for hate speech detection as part of the CREEP Social Media Analytics System for cyberbullying prevention. Cyberbullying is "an aggressive, intentional act carried out by a group or an individual, using electronic forms of contact, repeatedly and over time against a victim who cannot easily defend him or herself". In online social media, each episode of online activity aimed at offending, menacing, harassing or stalking another person can be classified as a cyberbullying phenomenon. This is connected even with concrete public health issues, since recent studies show that victims are more likely to suffer from psycho-social difficulties and affective disorders. Given its societal impact, the implementation of cyberbullying detection systems, combining abusive language detection and social network analysis, has attracted a lot of attention in the last years. In this context, together with the colleagues of Inria Rennes (partners of the CREEP project), we are investigating the potential of coupling text and images in the cyberbullying detection task, to consider also cases when the bullying phenomenon starts from a picture posted on a social media platform, or when an image is used to insult someone.

1.5 Structure of this report

This report is structured to describe my research contributions in each of the above-mentioned research areas. More precisely, in the following: Chapter 2 describes my research in information extraction to generate formal knowledge, according to different application scenarios; Chapter 3 presents my contribution in the conception and implementation of a Question Answering system over Linked Data; Chapter 4 provides my contribution in the area of Argument Mining; and, last but not least, Chapter 5 presents my contribution in the area of cyberbullying and abusive language detection. Conclusions summarize my research work, and discuss future perspectives.

[11] http://creep-project.eu/en/

Chapter 2

Information extraction to generate structured knowledge

This chapter summarizes my contributions related to the extraction of information and the acquisition of knowledge from text in different application scenarios (mainly robotics, music information retrieval and events extraction) to generate structured knowledge in a machine-readable and machine-interpretable format. The goal is to enable the automation of tasks such as smart content classification, integrated search, reasoning, and data-driven activities such as mining for patterns and trends, uncovering hidden relations and so on. These contributions fit the areas of Natural Language Processing and Semantic Web.

My research contributions on this topic have been published in several journal and venues. I provide below a list of the main publications on the topic, divided by application scenarios.

Mining common sense knowledge from the Web for robotics:

- Soufian Jebbara, Valerio Basile, Elena Cabrio, Philipp Cimiano (2019). Extracting common sense knowledge via triple ranking using supervised and unsupervised distributional models. Semantic Web 10(1): 139-158 [241]

- Jay Young, Lars Kunze, Valerio Basile, Elena Cabrio, Nick Hawes, Barbara Caputo (2017). Semantic web-mining and deep vision for lifelong object discovery. Proceedings of the International Conference on Robotics and Automation (ICRA-17): 2774-2779 [491])

- Jay Young, Valerio Basile, Lars Kunze, Elena Cabrio, Nick Hawes (2016). Towards Lifelong Object Learning by Integrating Situated Robot Perception and Semantic Web Mining. ECAI 2016: 1458-1466[489]

- Valerio Basile, Soufian Jebbara, Elena Cabrio, Philipp Cimiano (2016) Populating a Knowledge Base with Object-Location Relations Using Distributional Semantics. Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (EKAW-16): 34-50 [39]

Processing song lyrics:

- Michael Fell, Yaroslav Nachaev, Gabriel Meseguer Brocal, Elena Cabrio, Fabien Gandon, Geoffroy Peeters (2020). Lyrics Segmentation via Bimodal Text-audio Representation. To appear in Journal of Natural Language Engineering [174]

- Michael Fell, Elena Cabrio, Elmahdi Korfed, Michel Buffa, Fabien Gandon (2020). Love Me, Love Me, Say (and Write!) that You Love Me: Enriching the WASABI Song Corpus with Lyrics Annotations. Proceedings of the Language Resources and Evaluation Conference (LREC-2020): 2138-2147 [173]

- Michael Fell, Elena Cabrio, Fabien Gandon, Alain Giboin (2019). Song Lyrics Summarization Inspired by Audio Thumbnailing. Proceedings of the Recent Advances in Natural Language Processing conference (RANLP-19), 328-337 [172]

- Michael Fell, Elena Cabrio, Michele Corazza, Fabien Gandon (2019). Comparing Automated Methods to Detect Explicit Content in Song Lyrics. Proceedings of the Recent Advances in Natural Language Processing conference (RANLP-19), 338-344[171]

- Michael Fell, Yaroslav Nechaev, Elena Cabrio, Fabien Gandon (2018) Lyrics Segmentation: Textual Macrostructure Detection using Convolutions. Proceeding of the International Conference on Comuputational Linguistics (COLING 2018), 2044-2054[175]

Events Extraction from social media:.

- Amosse Edouard, Elena Cabrio, Sara Tonelli, Nhan Le Thanh (2017). Graph-based Event Extraction from Twitter. Proceedings of the Recent Advances in Natural Language Processing conference (RANLP-17) 222-230 [157]

- Amosse Edouard, Elena Cabrio, Sara Tonelli, Nhan Le Thanh (2017). You'll Never Tweet Alone: Building Sports Match Timelines from Microblog Posts. Proceedings of the Recent Advances in Natural Language Processing conference (RANLP-17), 214-221 [159]

- Amosse Edouard, Elena Cabrio, Sara Tonelli and Nhan Le Thanh (2017). Semantic Linking for Event-Based Classification of Tweet. Proceedings of the 18th International Conference on Computational Linguistics and Intelligent Text Processing (CICLing-17) [158]

The contributions reported in this chapter are the results of several collaborations in the contexts of the CHIST-ERA ALOOF project (for robotics), the ANR-WASABI project and PhD of Michael Fell (for Music Information Retrieval), and the Ph.D. thesis of Amosse Edouard (for events extraction).

The justification underlying the first line of work in the robotics scenario is given by the fact that autonomous robots that are to assist humans in their daily lives are required, among other things, to recognize and understand the meaning of task-related objects. However, given an open-ended set of tasks, the set of everyday objects that robots will encounter during their lifetime is not foreseeable. That is, robots have to learn and extend their knowledge about previously unknown objects on-the-job. We therefore propose multiple approaches to automatically acquire parts of this knowledge (e.g., the *class* of an object, its *typical location* and *manipulation-relevant* knowledge that can support robots' action planning) in form of ranked hypotheses from the Semantic Web using contextual information extracted from observations and experiences made by robots. Thus, by integrating situated robot perception and Semantic Web mining, robots can continuously extend their object knowledge beyond perceptual models which allows them to reason about task-related objects, e.g., when searching for them, robots can infer the most likely object locations or their typical use. The RDF publishing of the extracted knowledge base was also key to support (instantaneous) knowledge-sharing among different robots and even beyond. The Semantic Web and linked data approach was chosen to make the extracted/learned knowledge available to more than just one robot.

Goal of the second line of work is to understand how the automatic processing of song lyrics can be exploited to enhance applications such as music recommendation systems and music information retrieval. While most of the existing work in this area focus on audio and metadata, our work demonstrates that the information contained in song lyrics is another important factor that should be taken into account when designing these applications. As a result, we present the WASABI Song Corpus, a large corpus of songs enriched with metadata extracted from music databases on the Web, and resulting from the processing of song lyrics and from audio analysis. Given that lyrics encode an important part of the semantics of a song, we present multiple methods to extract relevant information from the lyrics, such as their structure segmentation, their topics, the explicitness of the lyrics content, the salient passages of a song and the emotions conveyed. The corpus contains 1.73M songs with lyrics (1.41M unique lyrics) annotated at different levels with the output of the above mentioned methods. Such corpus labels and the provided methods can be exploited by music search engines and music professionals (e.g., journalists, radio presenters) to better handle large collections of lyrics, allowing an intelligent browsing, categorization and segmentation recommendation of songs.

The third line of work arise from the observation that the capability to understand and analyze the stream of messages exchanged on social media (e.g., Twitter) is an effective way to monitor what people think, which trending topics are emerging, and which main events are affecting people's lives. This is crucial for companies interested in social media monitoring, as well as for public administrations and policy makers, that monitor tweets in order to report or confirm recent events. In this direction, in our work we address the task of analyzing tweets to discover new or track previously identified events. First, to classify event-related tweets, we explore the impact of entity linking and of the NEs generalization, and we apply a supervised classifier to separate event-related from non event-related tweets, as well as to associate to event-related tweets the event categories defined by the Topic Detection and Tracking community. Second, we address the task of detecting which tweets describe a specific event and cluster them. We propose a novel approach that exploits NE mentions in tweets and their local context to create a temporal event graph. Then, we process the event graphs to detect clusters of tweets describing the same event. Third, we propose an approach to build a timeline with actions in a sports game based on tweets, combining information provided by external knowledge bases to enrich the content of the tweets, and apply graph theory to model relations between actions and participants in a game.

This chapter is organized as follows: Sections 2.1 and 2.2 describe the work carried out to equip a robot with a database of object knowledge. Then, Sections 2.4, 2.5 and 2.6 present the methods we proposed to annotate relevant information in the song lyrics to enrich the WASABI Song corpus with lyrics metadata. Sections 2.7 and 2.8 explain the events extraction methods to classify and cluster events, and to create timelines. Conclusions end the chapter.

2.1 Towards lifelong object learning by integrating stuated robot perception and Semantic Web mining

It is crucial for autonomous robots working in human environments such as homes, offices or factories to have the ability to represent, reason about, and learn new information about the objects in their environment. Current robot perception systems must be provided with models of the objects in advance, and their extensibility is typically poor. This includes both perceptual models (used to recognize the object in the environment) and semantic models (describing what the object is, what it is used for etc.). Equipping a robot *a priori* with a (necessarily closed) database of object knowledge is problematic because the system designer must predict which

subset of all the different domain objects is required, and then build all of these models (a time-consuming task). If a new object appears in the environment, or an unmodelled object becomes important to a task, the robot will be unable to perceive, or reason about, it. The solution to this problem is for the robot to *learn on-line about previously unknown objects*. This allows robots to autonomously extend their knowledge of the environment, training new models from their own experiences and observations.

The online learning of perceptual and semantic object models is a major challenge for the integration of robotics and AI. In this section we address one problem from this larger challenge: given an observation of a scene containing an unknown object, can an autonomous system predict the semantic description of this object. This is an important problem because online-learnt object models [167] must be integrated into the robot's existing knowledge base, and a structured, semantic description of the object is crucial to this. Our solution combines semantic descriptions of perceived scenes containing unknown objects, with a distributional semantic approach which allows us to fill gaps in the scene descriptions by mining knowledge from the Semantic Web. Our approach assumes that the knowledge onboard the robot is a subset of some larger knowledge base, i.e. that the object is not unknown beyond the robot's pre-configured knowledge. To determine which concepts from this larger knowledge base might apply to the unknown object, our approach exploits the spatio-temporal context in which objects appear, e.g., a teacup is often found next to a teapot and sugar bowl. These spatio-temporal co-occurrences provide contextual clues to the properties and identity of otherwise unknown objects.

This section makes the following contributions:

- a novel distributional semantics-based approach for predicting both the semantic identity of an unknown, everyday object based on its spatial context and its most likely location based on semantic relatedness;
- an extension to an existing semantic perception architecture to provide this spatial context; and
- an evaluation of these techniques on real-world scenes gathered from a long-term autonomous robot deployment.

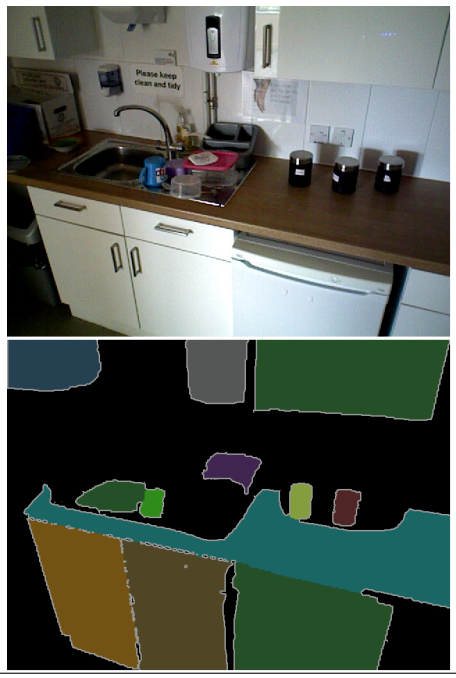
In the following, in Section 2.1.1, we first state the problem of acquiring semantic descriptions for unknown objects and give an overview of our approach. In Subsection 2.1.2, we describe the underlying robot perception system and explain how it is integrated with a Semantic Web mining component. Subsection 2.1.3 describes how the component generates answers/hypotheses to web-queries from the perception module. In Subsection 2.1.4, we describe the experimental setup and present the results, while in Subsection 2.1.4 we provide a detailed discussion about our approach.[1]

2.1.1 Problem statement and methodology

Problem Statement. The problem we consider in this section can be summarized as follows: *Given the context of a perceived scene and the experience from previous observations, predict the class of an 'unknown' identified object.* The context of a scene can include information about the types and locations of recognized small objects, furniture, and the type of the room where the observation has been made.

In this section we use the following running example (Figure 2.1) to illustrate the problem and our approach:

[1] We also make available our data set and software source code at: `http://github.com/aloof-project/`



Room	kitchen
Surface	counter-top
Furniture	refrigerator, kitchen cabinet, sink
Small Objects	bowl, teabox, instant coffee, water boiler, mug

Figure 2.1: Perceived and interpreted kitchen scene, with various objects.

> While operating 24/7 in an office environment, a robot routinely visits the kitchen and scans all surfaces for objects. On a kitchen counter it finds several household objects: a bowl, a teabox, a box of instant coffee, and a water boiler. However, one of the segmented objects, a mug, cannot be identified as one of the known object classes. The robot's task is to identify the unknown object solely based on the context of the perceived scene and scenes that have been previously perceived and in which the respective object was identified.

The problem of predicting the class of an object purely based on the context can also be seen as *top-down reasoning* or *top-down processing* of information. This stands in contrast to data-driven bottom-up processing where, for example, a robot tries to recognize an object based on its sensor data. In top-down processing, an agent, or the robot, has some expectations of what it will perceive based on commonsense knowledge and its experiences. For example, if a robot sees a fork and a knife close to each other, and a flat unknown object with a square bounding-box next to them, it might deduce that the unknown object is probably a plate. In the following, we refer to this kind of processing which combines top-down reasoning and bottom-up perception as *knowledge-enabled perception*.

Systems such as the one described in this section are key components of integrated, situated AI systems intended for life-long learning and extensibility. We currently develop the system with two main use-cases in mind, both stemming from the system's capability to suggest information about unknown objects based on the spatial context in which they appear. The first use case is as part of a crowd-sourcing platform, allowing humans that inhabit the robot's environment to help it label unknown objects. Here, the prediction system is used to narrow down the list of candidate labels and categories to be shown to users to select from alongside images of unknown objects the robot has encountered. Our second use case will be to help form more informative queries for larger machine learning systems, in our case an image classification

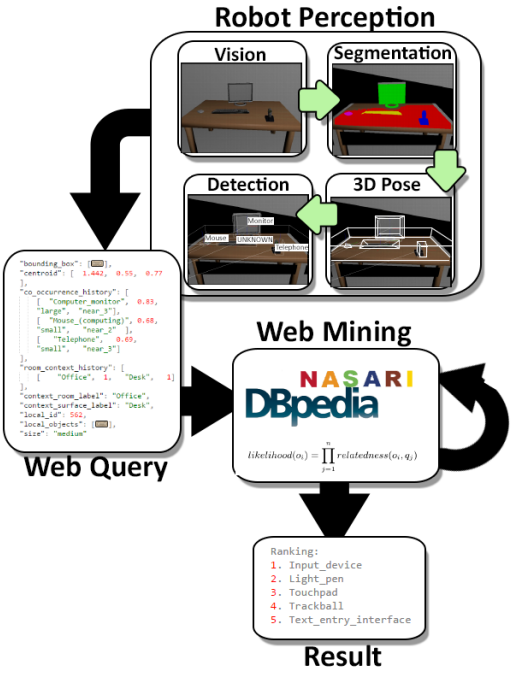


Figure 2.2: System overview. The robot perception component identifies all object candidates within a scene. All object candidates that can be recognized are labeled according to their class, all remaining objects are labeled as 'unknown'. Furthermore, the component computes the spatial relations between all objects in the scene. Together with context information from a semantic environment map, the robot generates a query to a web service which is processed by the Semantic Web mining component. Based on the semantic relatedness of objects the component provides a ranked list of the potential classes for all unknown objects.

system trained on extensive, though categorized, image data from websites like Amazon. Here, having some hints as to an object's identity, such as a distribution over a set of possible labels or categories it might belong to or be related to, could produce a significant speed boost by letting the classification system know what objects it does *not* have to test against. In this case, we aim to use the system to help a robot make smarter, more informed queries when asking external systems questions about the world.

Our approach. We address the problem of predicting information about the class of an object based on the perceived scene context by mining the Semantic Web. The extracted scene context includes a list of recognized objects and their spatial relations among each other, plus additional information from a semantic environment map. This information is then used to mine potential object classes based on the semantic relatedness of concepts in the Web. In particular, we use DBpedia as a resource for object knowledge, and will later on use WordNet to investigate object taxonomies. The result of the web mining component is a ranked list of potential objects classes, expressed as DBpedia entries, which allows us access to further information beyond just the class of an object, such as categorical knowledge. An overview of the entire developed system is given in Figure 2.2.

Overall, we see our context-based class prediction approach as a means to restrict the number of applicable classes for an object. The aim of our knowledge-enabled perception system is not to replace a bottom-up perception system but rather to complement it as an additional *expert*. For example, in the context of a crowdsourcing-based labeling platform our system could generate label suggestions for users. Thereby labeling tasks can be performed in less time and object labels would be more consistent across users. Hence, we believe that our

system provides an essential functionality in the context of lifelong object learning.

In the following, we briefly discuss various resources of object knowledge.

Resources for object knowledge. To provide a common format for object knowledge, and to access the wide variety of structured knowledge available on the Web, we link the observations made by the robot to DBpedia concepts. DBpedia [54] is a crowd-sourced community effort started by the Semantic Web community to extract structured information from Wikipedia and make this information available on the Web. DBpedia has a broad scope of entities covering different domains of human knowledge: it contains more than 4 million things classified in a consistent ontology and denoted by a URI-based reference of the form `http://dbpedia.org/page/Teapot` for the Teapot concept. DBpedia supports sophisticated queries (using an SQL-like query language for RDF called SPARQL) to mine relationships and properties associated with Wikipedia resources. We link the objects that the robot can encounter in natural environments to DBpedia concepts, thus exploiting this structured, ontological knowledge.

BabelNet [341] is both a multilingual encyclopedic dictionary and a semantic network which connects concepts and named entities in a very large network of semantic relations (about 14 million entries). BabelNet covers and is obtained from the automatic integration of several resources, such as WordNet [177], Wiktionary and Wikipedia. Each concept contained in BabelNet is represented as a vector in a high-dimensional geometric space in the NASARI resource, that we use to compute the semantic relatedness among objects.

2.1.2 Situated robot perception

The RoboSherlock framework. To be able to detect both known and unknown objects in its environment a robot must have perceptual capabilities. Our perception pipeline is based on the *RoboSherlock framework* [43], an open-source framework for implementing perception systems for robots, geared towards interaction with objects in human environments. The use of RoboSherlock provides us with a suite of vision and perception algorithms. Following the paradigm of Unstructured Information Management (as used by the IBM Watson project), RoboSherlock approaches perception as a problem of content analysis, whereby sensor data is processed by a set of specialized information extraction and processing algorithms called *annotators*. The RoboSherlock perception pipeline is a sequence of annotators which include plane segmentation, RGB-D object segmentation, and object detection algorithms. The output of the pipeline includes 3D point clusters, bounding-boxes of segmented objects (as seen in Figure 2.2), and feature vectors (color, 3D shape and texture) describing each object. These feature vectors are important as they allow the robot to track unknown objects as it takes multiple views of the same scene. Though in this section we work with a collected and annotated dataset, we do not require the segmentation or 3D object recognition steps RoboSherlock can provide via LINE-MOD-3D [223], though this component is used in our full Robot and Simulated system where a range of perception algorithms are connected and used instead of dataset input. We make use of all other RoboSherlock capabilities the pipeline to process the data and provide a general architecture for our representation and extraction of historical spatial context, web query generation and the application of Qualitative Spatial Relations, which we will discuss in a following section.

Scene perception. We assume here that the robot is tasked with observing objects in natural environments. Whilst this is not a service robot task in itself, it is a precursor to many other task-driven capabilities such as object search, manipulation, human-robot interaction etc. Similar to prior work (e.g., [407]) we assume that the robot already has a semantic map

of its environment which provides it with at least annotations of supporting surfaces (desks, worktops, shelves etc.), plus the semantic category of the area in which the surface is located (office, kitchen, meeting room etc.). Surfaces and locations are linked to DBpedia entries just as object labels are, typically as entities under the categories `Furniture` and `Room` respectively.

From here, we have access to object, surface and furniture labels described by the data, along with 3D bounding-boxes via 3D point data. In the kitchen scene the robot may observe various objects typical of the room, such as a refrigerator, a cabinet, mugs, sugar bowls or coffee tins. Their positions in space relative to a global map frame are recorded and we can then record the distance between objects, estimate their size (volume) and record information about their co-occurrences, and the surfaces upon which they were observed, by updating histograms attached to each object.

In the following we assume that each scene only contains a single unknown object, but the approach generalizes to multiple unknown objects treated independently. Joint inference over multiple unknown objects is future work.

Spatial and semantic context extraction. In order to provide additional information to help subsequent components predict the unknown object, we augment the scene description with additional spatial and semantic *context* information, describing the relationships between the unknown object and the surrounding known objects and furniture. This context starts from the knowledge we already have in the semantic map: labels for the room and surface the object is supported by.

We make use of *Qualitative Spatial Relations* (QSRs) to represent information about objects [187]. QSRs discretise continuous spatial measurements, particularly relational information such as the distance and orientation between points, yielding symbolic representations of ranges of possible continuous values. In this work, we make use of a qualitative distance measure, often called a Ring calculus. When observing an object, we categorize its distance relationship with any other objects in a scene with the following set of symbols: $near_0, near_1, near_2$, where $near_0$ is the closest. This is accomplished by placing sets of thresholds on the distance function between objects, taken from the centroid of the 3D cluster. For example, this allows us to represent that the mug is closer to the spoon than the kettle ($near_0(mug, spoon)$ $near_2(mug, kettle)$) without using floating-point distance values based on noisy and unreliable readings from the robot's sensors. The RoboSherlock framework provides a measure of the qualitative size of objects by thresholding the values associated with the volume of 3D bounding-boxes around objects as they are observed. We categorize objects as $small, medium, large$ in this way, allowing the robot to represent and compare object sizes. Whilst our symbolic abstractions are currently based on manual thresholds, approaches exist for learning parametrisations of QSRs through experience (e.g., [490]) and we will try this in the future. For now, we choose parameters for our qualitative calculi tuned by our own knowledge of objects in the world, and how they might relate. We use $near_0$ for distances in cluster space lower than 0.5, $near_1$ for distances between than 0.5 and 1.0, $near_2$ for distances between 1.0 and 3.5 and $near_3$ for distances greater than 3.5.

As the robot makes subsequent observations, it may re-identify the same unknown object in additional scenes. When this happens we store all the scene descriptions together, providing additional context descriptions for the same object. In Figure 2.3 we show part of the data structure describing the objects that co-occurred with a plate in a kitchen, and their *most common* qualitative spatial relations.

```
"co_occurrences": [
  ["Coffee",  0.5, "near_0" ],
  ["Kitchen_side", 1.0,"near_0"  ],
  ["Kitchen_cabinet", 1.0,"near_1" ],
  ["Fridge", 0.625,"near_1" ],
  ["Teabox", 0.625,"near_0" ],
  ["Waste_container",  0.375,"near_2" ],
  ["Utensil_rack", 0.625, "near_1" ],
  ["Sugar_bowl_(dishware)",  0.625,"near_0" ],
  ["Electric_water_boiler", 0.875,"near_1" ],
  ["Sink", 0.625, "near_1"  ] ],
   "context_history": [
     ["Kitchen", 1.0, "Kitchen_counter",1 ],
     [  "Office", 0.0,"Desk", 0 ]],
  "context_room_label": "Kitchen",
  "context_surface_label": "Kitchen_counter",
```

Figure 2.3: An example data fragment taken from a series of observations of a Plate in a series of kitchen scenes, showing object, furniture, room and surface co-occurrence

2.1.3 Semantic Web mining

For an unknown object, our aim is to be able to provide a list of likely DBpedia concepts to describe it, and we will later consider and compare the merits and difficulties associated with providing object *labels* and object *categories*. As this knowledge is not available on the robot (the object is *locally* unknown), it must query an external data source to fill this knowledge gap. We therefore use the scene descriptions and spatial contexts for an unknown object to generate a query to a Web service. In return this service provides a list of the possible DBpedia concepts which may describe the unknown object. We expect the robot to use this list in the future to either automatically label a new object model, or to use the list of possible concepts to guide a human through a restricted (rather than open-ended) learning interaction.

The Web service provides access to object- and scene-relevant knowledge extracted from Web sources. It is queried using a JSON structure sent via an HTTP request (shown in Figure 2.2). This structure aggregates the spatial contexts collected over multiple observations of the unknown object. In our current work we focus on the co-occurrence structure. Each entry in this structure describes an object that was observed with the unknown object, the ratio of observations this object was in, and the spatial relation that most frequently held between the two. The room and surface fields describe where the observations were made.

Upon receiving a query, the service computes the *semantic relatedness* between each object included in the co-occurrence structure and every object in a large set of candidate objects from which possible concepts are drawn from (we discuss the nature of this set later on).

This semantic relatedness is computed by leveraging the vectorial representation of the DBpedia concepts provided by the NASARI resource [96]. In NASARI each concept contained in the multilingual resource BabelNet [341] is represented as a vector in a high-dimensional geometric space. The vector components are computed with the *word2vec* [320] tool, based on the co-occurrence of the mentions of each concept, in this case using Wikipedia as source corpus.

Since the vectors are based on distributional semantic knowledge (based on the *distributional hypothesis*: words that occur together often are likely semantically related), vectors that represent related entities end up close in the vector space. We are able to measure such re-

latedness by computing the inverse of the cosine distance between two vectors. For instance, the NASARI vectors for `Pointing_device` and `Mouse_(computing)` have relatedness 0.98 (on a continuous scale from 0 to 1), while `Mousepad` and `Teabox` are 0.26 related.

The system computes the aggregate of the relatedness of a candidate object to each of the scene objects contained in the query. Using relatedness to score the likely descriptions of an unknown object follows from the intuition that related objects are more likely than unrelated objects to appear in a scene, e.g., to identify a `Teapot` is more useful to know that there is a `Teacup` at the scene rather than a `Desk`.

Formally, given n observed objects in the query $q_1, ..., q_n$, and m candidate objects in the universe under consideration $o_1, ..., o_m \in O$, each o_i is given a score that indicates its likelihood of being the unknown object by aggregating its relatedness across all observed objects. The aggregation function can be as simple as the arithmetic mean of the relatedness scores, or a more complex function. For instance, if the aggregation function is the product, the likelihood of an object o_i is given by:

$$likelihood(o_i) = \prod_{j=1}^{n} relatedness(o_i, q_j)$$

For the sake of this work, we experimented with the product as aggregating function. This way of aggregating similarity scores gives higher weight to highly related pairs, as opposed to the arithmetic mean, where each query object contributes equally to the final score. The idea behind this choice is that if an object is highly related to the target it should be regarded as more informative.

The information carried by each query is richer than just a bare set of object labels. One piece of knowledge that can be exploited to obtain a more accurate prediction is the relative position of the observed objects with respect to the target unknown object. Since this information is represented as a discrete level or proximity (from `near_0` to `near_3`), we can use this as a threshold to determine whether or not an object should be included in relatedness calculation. In this work we discard any object related by `near_3`, based on the intuition that the further away an object is spatially, the less related it is. The Results section includes an empirical investigation into approach. For clarity, here we present an example of execution of the algorithm described above on the query corresponding to the kitchen example seen throughout the section. The input to the Web module is a query containing a list of pairs (object, distance): (`Refrigerator`, 3), (`Kitchen_cabinet`, 3), (`Sink`, 3), (`Kitchen_cabinet`, 3), (`Sugar_bowl_(dishware)`, 1), (`Teabox`, 1), (`Instant_coffee`, 2), (`Electric_water_boiler`, 3). For the sake of readability, let us assume a set of candidate objects made only of three elements: `Tea_cosy`, `Pitcher_(container)` and `Mug`. Table 2.1 show the full matrix of pairwise similarities.

Among the three candidates, the one with highest aggregated score is `Tea_cosy`, followed by `Mug` and `Pitcher_(container)`. For reference, the ground truth in the example query is `Mug`, that ended up second in the final ranking returned by the algorithm.

We can also alter the performance of the system using the *frequency* of the objects returned by the query. The notion of frequency, taken from [133], is a measure based on the number of incoming links in the Wikipedia page of an entity. Using this measure we can choose to filter uncommon objects from the results of the query, by thresholding with a given frequency value. In the example above, the frequency counts of `Tea_cosy`, `Pitcher_(container)` and `Mug` are respectively 25, 161 and 108. Setting a threshold anywhere between 25 and 100 would filter `Tea_cosy` out of the result, moving up the ground truth to rank 1. Similarly, we can filter out objects that are too far from the target by imposing a limit on their observed distance. A threshold of 2 (inclusive) for the distance of the objects in the example would exclude `Refrigerator`, `Kitchen_cabinet`, `Sink` and `Electric_water_boiler` from the computation.

	Tea_cosy	Pitcher_(container)	Mug
Refrigerator	0.473	0.544	0.522
Sink	0.565	0.693	0.621
Sugar_bowl_(dishware)	0.555	0.600	0.627
Teabox	0.781	0.466	0.602
Instant_coffee	0.821	0.575	0.796
Electric_water_boiler	0.503	0.559	0.488
product	**0.048**	**0.034**	**0.047**

Table 2.1: Object similarity of the three candidates `Tea_cosy`, `Pitcher_(container)` and `Mug` to the objects observed at the example kitchen scene. The last line shows the similarity scores aggregated by product.

Other useful information available from the spatial context includes the label of the room, surface or furniture where the unknown was observed. Unfortunately, in order to leverage such information, one needs a complete knowledge base containing these kind of relations, and such a collection is unavailable at the moment. However, the room and the surface labels are included in the relatedness calculations along with the observed objects.

2.1.4 Experiments

In order to evaluate the effectiveness of the method we propose in predicting unknown objects' labels, we perform some experimental tests. In this section we report on the experimental setup and the results we obtained, before discussing them in further detail.

Experimental Set-up. Our experimental evaluation is an experiment based on a collection of panoramic RGB-D scans taken from an autonomous mobile service robot deployed in a working office for a month. It took these scans at fixed locations according to a flexible schedule. After the deployment we annotated the objects and furniture items in these sweeps, providing each one with a DBpedia concept. This gives us 1329 real world scenes (384 kitchen, 945 office) on which we can test our approach. From this data, our evaluation treats each labeled object in turn as an unknown object in a leave-one-out experiment, querying the Web service with the historical spatial context data for the unknown object similar to that shown in Figure 2.3.

Figure 2.4: An example office scene as an RGB image from our real-world deployment. Our data contains 945 office scenes, and 384 kitchen scenes.

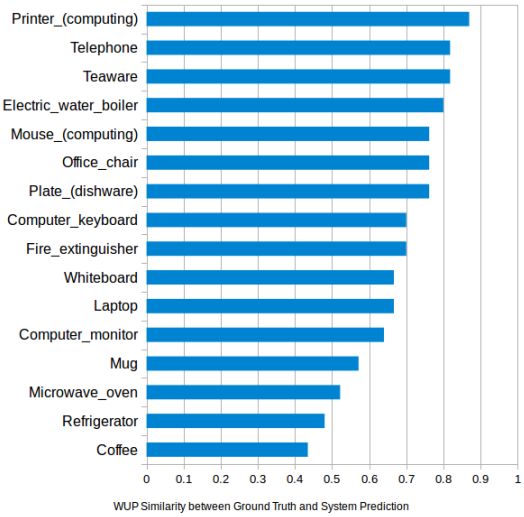


Figure 2.5: WUP similarity measure between WordNet synsets of ground truth and top-ranked result, with $t = 50$, $p = 2$ using the *prod* method. Ranks closer to 1 are better. Values closer to 1 indicate similarity, and values closer to 0 indicate dissimilarity.

In all of the experiments we compare the ground truth (known label in the data) to the DBpedia concepts predicted by our system. We measure performance based on two metrics. The first *WUP similarity* measures the semantic similarity between the ground truth and the concept predicted as most likely for the unknown object. The second measure is the *ranking* of the ground truth in the list of suggested concepts.

For the experiments, the set of candidate objects (O in Section 2.1.3) was created by adding all concepts from the DBpedia ontology connected to the room types in our data by up to a depth of 3. For example, starting from office leads us to office equipment, computers, stationary etc. This resulted in a set of 1248 possible concepts. We set the frequency threshold to 20, meaning we ignored any suggest concept which had a frequency value lower than this. This means uncommon concepts such as `Chafing_dish` (frequency=13) would always be ignored if suggested, but more common ones such as `Mouse_(computing)` (frequency=1106) would be kept.

Results. Figure 2.5 shows the result of calculating the WUP similarity [479] between the WordNet synsets of the ground truth and the top-ranked result from our semantic web-mining system. WUP measures semantic relatedness by considering the depth of two synsets in addition to the depth of their Lowest Common Subsumer (LCS). This means that large leaps between concepts will reduce the eventual similarity score more than small hops might. To do this we used ready available mappings to link DBpedia concepts in our system to WordNet synsets, which are themselves organized as a hierarchy of *is-a* relations. This is in contrast to DBpedia, which is organized as a directed acyclic graph, and while that still means that we could apply the WUP measure to DBpedia nodes directly, WordNet offers a more structured taxonomy of concepts that is more well-suited to this kind of work. This serves to highlight the importance of a multi-modal approach to the use of such ontologies. In the results, the system predicted *Lightpen* when the ground truth was *Mouse* producing a WUP score of 0.73, with the LCS being the *Device* concept, with *Mouse* and *Lightpen* having depth 10 and 11 respectively, and *Device* having depth 8 measured from the root node of *Entity*. In this case, the system suggested an object that fell within 3 concepts of the ground truth, and this is true for the majority of the results in Figure 2.5. However, in the case of *refrigerator* as the ground truth,

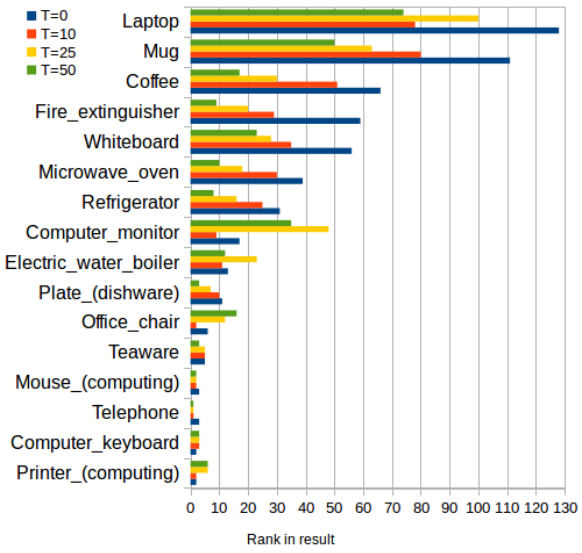


Figure 2.6: Rank in result by object *category*, matching the highest ranked object with a category shared with the ground truth in the result set, with varying values of the parameter t, with $p = 2$ and the *prod* method. Ranks closer to 1 are better. Ranking is determined by the position in the result of the first object with an immediate category in common with the ground truth. 56% (9/16) achieve <= 10.

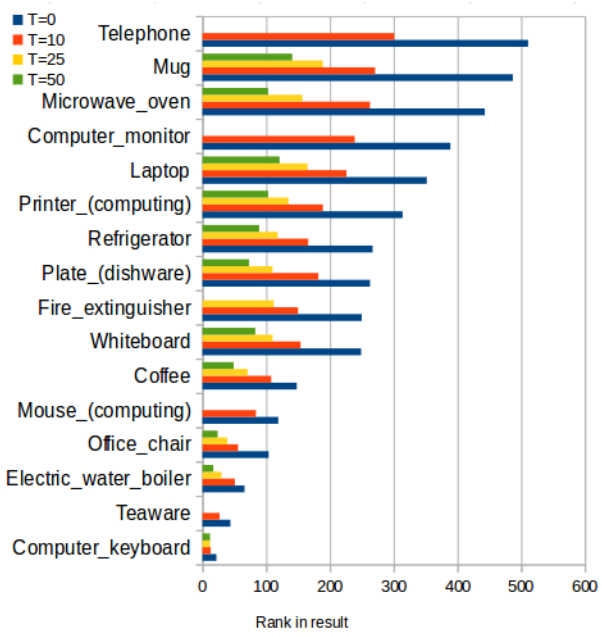


Figure 2.7: Rank in result by object *label*, matching the label of the ground truth in the result set, with varying values of the parameter t, with $p = 2$ and the *prod* method. Increasing values of T can cause some objects to be excluded from the result set entirely, such as the Teaware or Monitor at T=50

	Mean	Median	Std. Dev	Variance	Range
WUP	0.69	0.70	0.12	0.01	0.43
Category Rank	17.00	9.50	20.17	407.20	73.00
Object Rank	50.93	36.5	50.18	2518.32	141

Figure 2.8: Statistics on WUP and Rank-in-result data, both for $t = 50, p = 2$ using *prod*

the system suggests *keypad* as the highest ranked result, producing a WUP score of 0.52. Here, the LCS is at depth 6 with the concept $Artifact$, the ground truth $refrigerator$ is at depth 13 and the prediction *keypad* is at depth 10. While in this case the node distance between the LCS and the prediction is 4, where in the previous example it was 3, the WUP score is much worse here (0.73 vs 0.52) as there are more large leaps across conceptual space. Our best result in this experiment is for *Printer* as the ground truth, for which the system suggests *keypad* again, however the LCS here is the *peripheral* node at depth 10, where *printer* is at depth 11 and *keypad* is at depth 12.

The system suggests a range of objects that are closely related to the unknown object, inferred only from its spatial context and knowledge of the objects and environment around it. From here this allows us to generate a list of candidate concepts which we can use in a second stage of refinement, such as by presentation to a human-in-loop.

Figure 2.6 shows how frequency thresholding effects the performance of the system. In this experiment we consider the position in the ranked result of the first object with an immediate parent DBpedia category in common with the ground truth. Doing so essentially maps the larger set of object labels to a smaller set of object categories. This is in contrast to considering the position in the result of the specific ground truth label, as shown in Figure 2.7, and allows us to generate a ranking over *categories of objects*. To ensure categories remain relevant to the situated objects we are interested in, we prune a number DBpedia categories such as those listing objects invented in certain years, or in certain countries. We regard these as being overly broad, and provide a more abstract degree of semantic knowledge about objects than we are interested in. As such, we retrieve the rank-in-result of the first object that shares an immediate DBpedia category with the ground truth, which in the case of *Electric water boiler* turns out to be *Samovar*, a kind of Russian water boiler, as both share the immediate ancestor category $Boilers\_(cookware)$. The *Samovar*, and thus the boiler category, appears at rank 12, whereas the specific label *Electric water boiler* appears near the end of the result set of 1248 objects, which covers 641 unique DBpedia categories. In our results, categories associated with 9 of the 16 objects (56%) appear within the result's top 10 entries. Here as we filter out uncommon words by increasing the filter threshold T we improve the position of the concept in the list. Whilst this allows us to definitely remove very unlikely answers that appear related due to some quirk of the data, the more we also start to reduce the ability of the robot to learn about certain objects. This is discussed further in the Discussion paragraph below.

Unlike WordNet synsets and concepts, DBpedia categories are more loosely defined and structured, being generated from Wikipedia, but this means they are typically richer in the kind of semantic detail and broad knowledge representation that may be more suitable for presentation to humans, or more easily mapped to human-authored domains. While WordNet affords us access to a well-defined hierarchy of concepts, categories like *device* and *container* are fairly broad, whereas DBpedia categories such as $Video\_game\_control\_methods$ or $Kitchenware$ describe a smaller set of potential objects, but may be more semantically meaningful when presented to humans.

Discussion. Overall, whilst the results of our object category prediction system show that it is possible for this novel system to generate some good predictions, the performance is variable across objects. There are a number of factors that influence performance, and lead to this variability. The first issue is that the current system does not rule out suggestions of things it already knows. For example if the unknown object is a keyboard, the spatial context and relatedness may result in a top suggestion of a mouse, but as the system already knows about that, it is probably a less useful suggestion. However, it is possible that the unknown object could be a mouse, but has not been recognized correctly. Perhaps the most fundamental issue in the challenge of predicting objects concepts from limited information is how the limit the scope of suggestions. In our system we restricted ourselves to 1248 possible concepts, automatically selected from DBpedia by ontological connectivity. This is clearly a tiny fraction of all the possible objects in existence. On one hand this means that our autonomous robot will potentially be quite limited in what it can learn about. On the other hand, a large number of this restricted set of objects still make for highly unlikely suggestions. One reason for this is the corpus-based automatically-extracted nature of DBpedia, which means that it includes interesting objects which may never be observed by a robot (e.g., `Mangle_(machine)`). More interestingly though is the effect that the structure of the ontology has on the nature of suggestions. In this work we have been using hierarchical knowledge to unpin our space of hypotheses (i.e. the wider world our robot is placed within), but have not addressed this within our system. This leads to a mismatch between our expectations and the performance of the system with respect to arbitrary precision. For example, if the robot sees a joystick as an unknown object, an appropriate DBpedia concept would seem (to us) to be `Controller_(computing)` or `Joystick`. However, much more specific concepts such as `Thrustmaster` and `Logitech_Thunderpad_Digital` are also available to the system in its current form. When learning about an object for the first time, it seems much more useful for the robot to receive a suggestion of the former kind (allowing it to later refine its knowledge to locally observable instances) than the latter (which unlikely to match the environment of the robot). Instead, returning the *category* of the ranked objects our system suggests allows us to go some way towards this as shown in Figure 2.6, but still provides us a range of possible candidate categories – though narrowed down from 641 possible categories, to in some cases less than 5. As such, from here we can switch to a secondary level of labeling: that of a human-in-loop. We will next integrate the suggestion system with a crowd-sourcing platform, allowing humans that inhabit the robot's environment to help it label unknown objects, as shown in the next Section. The suggestion system will be used to narrow down the list of candidate categories that will be shown to users as they provide labels for images of objects the robot has seen and learned, but has not yet labeled. While further work is necessary to refine the current 56% of objects that have a category in the top-10 ranked result, we expect that the current results will be sufficient enough to allow a human to pick a good label when provided a brief list of candidates and shown images of the unknown objects. Such systems are crucial for life-long situated learning for mobile robot platforms, and will allow robot systems to extend their world models over time, and learn new objects and patterns.

The issue of how to select which set of possible objects to draw suggestions from is at the heart of the challenge of this work: make the set too large and it is hard to get good, accurate suggestions, but make it too small and you risk ruling out objects that your robot may need to know about. Whilst the use of frequency-based filtering improved our results by removing low-frequency outliers, more semantically-aware approaches may be necessary to improve things further. Further improvements can be made, for instance we largely do not use current instance observations about the object, but prefer its historical context when forming queries. This may be the wrong thing to do in some cases, in fact it may be preferable to weight observations of object context based on their recentness. The difference between historical context and the

context of an object in a particular instance may provide important contextual clues, and allow us to perform other tasks such as anomaly detection or boost the speed of object search tasks.

One issue we believe our work highlights is the need to integrate a multi-modal approach to the use of differing corpora and ontologies. For instance, the more formal WordNet hierarchy was used to calculate the semantic relatedness of our experiment results, rather than the less formal DBpedia ontology. However we hold that the DBpedia category relationships are more useful in the human-facing component of our system. There exist other ontologies such as YAGO which integrates both WordNet and DBpedia, along with its own category system, that will certainly be of interest to us in the future as we seek to improve the performance of our system. One of our primary goals is to better exploit the hierarchical nature of these ontologies to provide a way of retrieving richer categorical information about objects. While reliably predicting the specific object label from spatial context alone is difficult, we *can* provide higher-level ancestor categories that could be used to spur further learning or improve previous results. As such, we view the prediction process as one of matching the characteristics of a series of increasingly more specific categories to the characteristics of an unknown object, rather than immediately attempting to match the specific lowest-level characteristics and produce the direct object label. This requires an ontology both formally-defined enough to express a meaningful hierarchy of categories for each item, *and* broad enough to provide us mapping to a large set of common-sense categories and objects. It is not clear yet which combination of existing tools will provide the best route to accomplishing this.

2.2 Mining semantic knowledge from the Web

As discussed before, embodied intelligent systems such as robots require world knowledge to reason on top of their perception of the world in order to decide which actions to take. Consider now another example, i.e. that of a robot having the task to tidy up an apartment by storing all objects in their appropriate place. In order to perform this task, a robot would need to understand where the "correct" or at least the "prototypical" location for each object is in order to come up with an overall plan on which actions to perform to reach the goal of having each object stored in its corresponding location.

In general, in manipulating objects, robots might have questions such as the following:

- *Where should a certain object typically be stored?*
- *What is this object typically used for?*
- *Do I need to manipulate a certain object with care?*

The answers to these questions require common sense knowledge about objects, in particular prototypical knowledge about objects that, in absence of abnormal situations or specific contextual conditions or preferences, can be assumed to hold.

In this article, we are concerned with extracting such common sense knowledge from a combination of unstructured and semi-structured data. We are in particular interested in extracting default knowledge, that is prototypical knowledge comprising relations that typically hold in 'normal' conditions [306]. For example, given no other knowledge, in a normal situation, we could assume that milk is typically stored in the kitchen, or more specifically in the fridge. However, if a person is currently having breakfast and eating cornflakes at the table in the living room, then the milk might also be temporarily located in the living room. In this sense, inferences about the location of an object are to be regarded as non-monotonic inferences that can be retracted given some additional knowledge about the particular situation. We model such default, or prototypical, knowledge through a degree of prototypicality, that is, we do not

claim that the kitchen is 'the prototypical location' for the milk, but instead we model that the degree of prototypicality for the kitchen being the default location for the milk is very high. This leads naturally to the attempt to computationally model this degree of prototypicality and rank locations or uses for each object according to this degree of prototypicality. We attempt to do so following two approaches. On the one hand, we follow a distributional approach and approximate the degree of prototypicality by the cosine similarity measure in a space into which entities and locations are embedded. We experiment with different distributional spaces and show that both semantic vector spaces as considered within the NASARI approach as well as embedded word representations computed on unstructured texts as produced by predictive language models such as skip-grams provide already a reasonable performance on the task. A linear combination of both approaches has the potential to improve upon both approaches in isolation. We have presented this approach before including empirical results for the `locatedAt` relation mentioned above in previous work [39]. As a second approach to approximate the degree of prototypicality, we use a machine learning approach trained on positive and negative examples using a binary classification scheme. The machine learning approach is trained to produce a score that measures the compatibility of a given pair of object and location/use in terms of their prototypicality. We compare these two approaches in this section, showing that the machine learning approach does not perform as well as expected. Contrary to our intuitions, the unsupervised approach relying on cosine similarity in embedding space represents a very strong baseline difficult to beat.

The prototypical knowledge we use to train and evaluate the different methods is on the one hand based on a crowdsourcing experiment in which users had to explicitly rate the prototypicality of a certain location for a given object. On the other hand, we also use extracted relations from ConceptNet and the SUN database [483]. Objects as well as candidate locations, or candidate uses in the case of the instrumental relation, are taken from DBpedia. While we apply our models to known objects, locations and uses, our model could also be applied to candidate objects, locations and uses extracted from raw text.

We have different motivations for developing such an approach to extract common sense knowledge from unstructured and semi-structured data.

First, from the point of view of cognitive robotics [292] and cognitive development, acquiring common sense knowledge requires many reproducible and similar experiences from which a system can learn how to manipulate a certain object. Some knowledge can arguably even not be acquired by self experience as relevant knowledge also comprises the mental properties that humans ascribe to certain objects. Such mental properties that are not intrinsic to the physical appearance of the object include for instance the intended use of an object. There are thus limits to what can be learned from self-guided experience with an object. In fact, several scholars have emphasized the importance of cultural learning, that is of a more direct transmission of knowledge via communication rather than self-experience. With our approach we are simulating such a cultural transmission of knowledge by allowing cognitive systems, or machines in our case, to acquire knowledge by 'reading' texts. Work along these lines has, for instance, tried to derive plans on how to prepare a certain dish by machine reading descriptions of household tasks written for humans that are available on the Web [437]. Other work has addressed the acquisition of scripts from the Web [392].

Second, while there has been a lot of work in the field of information extraction on extracting relations, the considered relations differ from the ones we investigate in this work. Standard relations considered in relation extraction are: *is-a, part-of, succession, reaction, production* [360, 81] or *relation, parent/child, founders, directedBy, area_served, containedBy, architect, etc.* [395], or *albumBy, bornInYear, currencyOf, headquarteredIn, locatedIn, productOf, teamOf* [57]. The literature till the time of this work had focused on relations that are of a factual nature and explicitly mentioned in the text. In contrast, we are concerned with relations that

are *i)* typically not mentioned explicitly in text, and *ii)* they are not of a factual nature, but rather represent default or prototypical knowledge. These are thus quite different tasks.

We present and compare different approaches to collect manipulation-relevant knowledge by leveraging textual corpora and semi-automatically extracted entity pairs. The extracted knowledge is of symbolic form and represented as a set of (`Subject, Relation, Object`) triples. While this knowledge is not physically grounded [213], this model can still help robots or other intelligent systems to decide on how to act, support planning and select the appropriate actions to manipulate a certain object.

2.2.1 Extraction of relations by a ranking approach based on distributional representations

This section presents our framework to extract relations between pairs of entities for the population of a knowledge base of manipulation-relevant data. We frame the task of relation extraction between entities as a ranking problem as it gives us great flexibility in generating a knowledge base that balances between coverage and confidence. Given a set of triples (s, r, o), where s is the subject entity, r the relation (or predicate) and o the object entity[2], we want to obtain a ranking of these triples. The produced ranking of triples should reflect the degree of prototypicality of the objects with respect to the respective subjects and relations.

Our general approach to produce these rankings is to design a scoring function $f(s, r, o)$ that assigns a score to each triple, depending on s, r, and o. The scoring function is designed in such a way that prototypical triples are assigned a higher score than less prototypical triples. Sorting all triples by their respective scores produces the desired ranking. With a properly chosen function $f(s, r, o)$, it is possible to extract relations between entities to populate a knowledge base. This is achieved by scoring candidate triples and inserting or rejecting them based on their respective scores, e.g., if the score is above a certain threshold.

In this work, we present different scoring functions and evaluate them in the context of building a knowledge base of common sense triples. All of our proposed approaches rely on distributional representations of entities (and words). We investigate different vector representations and scoring functions, all with different strengths and weaknesses.

Word space models (or distributional space models, or word vector spaces) are abstract representations of the meaning of words, encoded as vectors in a high-dimensional space. Traditionally, a word vector space is constructed by counting *co-occurrences* of pairs of words in a text corpus, building a large square n-by-n matrix where n is the size of the vocabulary and the cell i, j contains the number of times the word i has been observed in co-occurrence with the word j. The i-th row in a co-occurrence matrix is an n-dimensional vector that acts as a *distributional* representation of the i-th word in the vocabulary. The similarity between two words is geometrically measurable with a metric such as the cosine similarity, defined as the cosine of the angle between two vectors:

$$similarity(\vec{x}, \vec{y})_{cos} = \frac{\vec{x} \cdot \vec{y}}{\| \vec{x} \| \| \vec{y} \|}$$

This is the key point to linking the vector representation to the idea of semantic relatedness, as the *distributional hypothesis* states that "words that occur in the same contexts tend to have similar meaning" [216]. Several techniques can be applied to reduce the dimensionality of the co-occurrence matrix. Latent Semantic Analysis [262], for instance, uses Singular Value Decomposition to prune the less informative elements while preserving most of the topology of the vector space, and reducing the number of dimensions to 100-500.

[2]Here we use the terminology `subject` and `object` from the Semantic Web literature instead of the terminology `head` and `tail` that is typically found in relation extraction literature.

Recently, neural network based models have received increasing attention for their ability to compute dense, low-dimensional representations of words. To compute such representation, called *word embeddings*, several models rely on huge amounts of natural language texts from which a vector representation for each word is learned by a neural network. Their representations of the words are therefore based on *prediction* as opposed to *counting* [30].

Vector spaces created on word distributional representations have been successfully proven to encode word similarity and relatedness relations [388, 394, 117], and word embeddings have proven to be a useful feature in many natural language processing tasks [121, 266, 149] in that they often encode semantically meaningful information of a word.

We argue that it is possible to extract interaction-relevant relations between entities, e.g. (`Object, locatedAt, Location`), using appropriate entity vectors and the cosine similarity since the domain and range of the considered relations are sufficiently narrow. In these cases, the semantic relatedness might be a good indicator for a relation.

Ranking by cosine similarity and word embeddings. In the beginning of this section, we motivated the use of distributional representations for the extraction of relations in order to populate a database of common sense knowledge. As outlined, we frame the relation extraction task as a ranking problem of triples (s, r, o) and score them based on a corresponding set of vector representations $\boldsymbol{V}$ for subject and object entities.

In this section, we propose a neural network-based word embedding model to obtain distributional representations of entities. By using the relation-agnostic cosine similarity[3] as our scoring function $f(s, r, o) = similarity_{cos}(\vec{v}_s, \vec{v}_o)$, with $\vec{v}_s, \vec{v}_o \in \boldsymbol{V}$, we can interpret the vector similarity as a measure of semantic relatedness and thus as an indicator for a relation between the two entities.

Many word embedding methods encode useful semantic and syntactic properties [251, 326, 322] that we leverage for the extraction of prototypical knowledge. In this work, we restrict our experiments to the skip-gram method [320]. The objective of the skip-gram method is to learn word representations that are useful for predicting context words. As a result, the learned embeddings often display a desirable linear structure [326, 322]. In particular, word representations of the skip-gram model often produce meaningful results using simple vector addition [322]. For this work, we trained the skip-gram model on a corpus of roughly 83 million Amazon reviews [305].

Motivated by the compositionality of word vectors, we derive vector representations for the entities as follows: considering a DBpedia entity[4] such as `Public_toilet`, we obtain the corresponding label and clean it by removing parts in parenthesis, if any, convert it to lower case, and split it into its individual words. We retrieve the respective word vectors from our pretrained word embeddings and sum them to obtain a single vector, namely, the vector representation of the entity: $\vec{v}_{Public\_toilet} = \vec{v}_{public} + \vec{v}_{toilet}$. The generation of entity vectors is trivial for "single-word" entities, such as `Cutlery` or `Kitchen`, that are already contained in our word vector vocabulary. In this case, the entity vector is simply the corresponding word vector. By following this procedure for every entity in our dataset, we obtain a set of entity vectors $\boldsymbol{V}_{sg}$, derived from the original skip-gram word embeddings. With this derived set of entity vector representations, we can compute a score between pairs of entities based on the chosen scoring function, the cosine vector similarity[5]. Using the example of `locatedAt`-pairs,

[3] We also experimented with APSyn [404] as an alternative similarity measure which, unfortunately, did not work well in our scenario.

[4] For simplicity, we only use the local parts of the entity URI, neglecting the namespace `http://dbpedia.org/resource/`

[5] For any entity vector that can not be derived from the word embeddings due to missing vocabulary, we assume a similarity of -1 to every other entity.

this score is an indicator of how typical the location is for the object. Given an object, we can create a ranking of locations with the most prototypical location candidates at the top of the list (see Table 2.2). We refer to this model henceforth as `SkipGram/Cosine`.

Table 2.2: Locations for a sample object, extracted by computing cosine similarity on skip-gram-based vectors.

Object	**Location**	**Cos. Similarity**
Dishwasher	Kitchen	.636
	Laundry_room	.531
	Pantry	.525
	Wine_cellar	.519

Ranking by cosine similarity and semantically-aware entity representations. Vector representations of words are attractive since they only require a sufficiently large text corpus with no manual annotation. However, the drawback of focusing on words is that a series of linguistic phenomena may affect the vector representation. For instance, a polysemous word as *rock* (stone, musical genre, metaphorically strong person, etc.) is represented by a single vector where all the senses are conflated.

NASARI [96], a resource containing vector representations of most of DBpedia entities, solves this problem by building a vector space of concepts. The NASARI vectors are actually distributional representations of the entities in BabelNet [341], a large multilingual lexical resource linked to WordNet, DBpedia, Wiktionary and other resources. The NASARI approach collects co-occurrence information of concepts from Wikipedia and then applies a cluster-based dimensionality reduction. The context of a concept is based on the set of Wikipedia pages where a mention of it is found. As shown by Camacho-Collados et al. [96], the vector representations of entities encode some form of semantic relatedness, with tests on a sense clustering task showing positive results. Table 2.3 shows a sample of pairs of NASARI vectors together with their pairwise cosine similarity ranging from -1 (opposite direction, i.e. unrelated) to 1 (same direction, i.e. related).

Table 2.3: Examples of cosine similarity computed on NASARI vectors.

	Cherry	Microsoft
Apple	.917	.325
Apple_Inc.	.475	.778

Following the hypothesis put forward in the beginning of this section, we focus on the extraction of interaction-relevant relations by computing the cosine similarities of entities. We exploit the alignment of BabelNet with DBpedia, thus generating a similarity score for pairs of DBpedia entities. For example, the DBpedia entity `Dishwasher` has a cosine similarity of .803 to the entity `Kitchen`, but only .279 with `Classroom`, suggesting that the prototypical location for a generic dishwasher is the kitchen rather than a classroom. Since cosine similarity is a graded value on a scale from -1 to 1, we can generate, for a given object, a ranking of candidate locations, e.g., the rooms of a house. Table 2.4 shows a sample of object-location pairs of DBpedia labels, ordered by the cosine similarity of their respective vectors in NASARI. Prototypical locations for the objects show up at the top of the list as expected, indicating a relationship between the semantic relatedness expressed by the cosine similarity of vector representations and the actual locative relation of entities. We refer to this model as `NASARI/Cosine`.

Table 2.4: Locations for a sample object, extracted by computing cosine similarity on NASARI vectors.

Object	Location	Cos. Similarity
Dishwasher	Kitchen	.803
	Air_shower_(room)	.788
	Utility_room	.763
	Bathroom	.758
	Furnace_room	.749
Paper_towel	Air_shower_(room)	.671
	Public_toilet	.634
	Bathroom	.632
	Mizuya	.597
	Kitchen	.589
Sump_pump	Furnace_room	.699
	Air_shower_(room)	.683
	Basement	.680
	Mechanical_room	.676

Ranking by a trained scoring function. In the previous sections, we presented models of semantic relatedness for the extraction of relations. The employed cosine similarity function of these models is relation-agnostic, that is, it only measures whether there is a relation between two entities but not which relation in particular. The question that naturally arises is: *Instead of using a single model that is agnostic to the relation, can we train a separate model for each relation in order to improve the extraction performance?* We try to answer this question by introducing a new model, based on supervised learning.

To extend the proposed approach to any kind of relation we modify the first model we presented (i.e., ranking by cosine similarity and Word Embeddings) by introducing a parameterized scoring function. This scoring function replaces the cosine similarity which was previously employed to score pairs of entities (e.g., Object-Location). By tuning the parameters of this new scoring function in a data-driven way, we are able to predict scores with respect to arbitrary relations.

We define the new scoring function $f(s, r, o)$ as a bilinear form:

$$f(s, r, o) = tanh(\vec{v}_s^{\top} \boldsymbol{M}_r \vec{v}_o + b_r) \tag{2.1}$$

where $\vec{v}_s, \vec{v}_o \in \boldsymbol{V} \subseteq \mathbb{R}^d$ are the corresponding embedding vectors for the subject and object entities s and o, respectively, b_r is a bias term, and $\boldsymbol{M}_r \in \mathbb{R}^{d \times d}$ is the scoring matrix corresponding to the relation r. Our scoring function is closely related to the ones proposed by Jenatton et al. [242] as well as Yang et al. [485], however, we make use of the *tanh* activation function to map the scores to the interval $(-1, 1)$. In part, this relates to the Neural Tensor Network proposed by Socher et al. [417]. By initializing $\boldsymbol{M}_r$ as the identity matrix and b_r with 0, the inner term of the scoring function corresponds initially to the dot product of $\vec{v}_s$ and $\vec{v}_o$ which is closely related to the originally employed cosine similarity.

In order to learn the parameters $\boldsymbol{M}_r$ and b_r of the scoring function, we follow a procedure related to Noise Contrastive Estimation [329] and Negative Sampling [322] which is also used in the training of the skip-gram embeddings. This method uses "positive" and "negative" triples, $\mathcal{T}^+_{train}$ and $\mathcal{T}^-_{train}$, to iteratively adapt the parameters. The positive triples $\mathcal{T}^+_{train}$ are triples that truly express the respective relation. In our case, these triples are obtained by crowdsourcing and leveraging other resources (see Section 2.2.2). Given these positive triples, the set of corrupted negative triples $\mathcal{T}^-_{train}$ is generated in the following way: we generate negative triples

(s', r, o) and (s, r, o') for each positive triple $(s, r, o) \in \mathcal{T}^+$ by selecting negative subject and object entities s' and o' randomly from the set of all possible subjects and objects, respectively. The exact number of negative triples that we generate per positive triple is a hyper-parameter of the model which we set to 10 triples[6] for all our experiments.

The training of the scoring function is framed as a classification where we try to assign scores of 1 to all positive triples and scores of -1 to (randomly generated) negative triples. We employ the mean squared error (MSE) as the training objective:

$$\mathcal{L} = \frac{1}{N}\Bigg(\sum_{(s,r,o)\in\mathcal{T}^+_{train}} (1 - f(s,r,o))^2 + \sum_{(s,r,o)\in\mathcal{T}^-_{train}} (-1 - f(s,r,o))^2 \Bigg) \tag{2.2}$$

where $N = |\mathcal{T}^+_{train}| + |\mathcal{T}^-_{train}|$ is the size of the complete training set. During training, we keep the embedding vectors $\boldsymbol{V}$ fixed and only consider $\boldsymbol{M}_r$ and b_r as trainable parameters to measure the effect of the scoring function in isolation. Presumably, this allows for a better generalization to previously unseen entities.

Due to the moderate size of our training data, we regularize our model by applying Dropout [422] to the embedding vectors of the head and tail entity. We set the dropout fraction to 0.1, thus only dropping a small portion of the 100 dimensional input vectors.

The supervised model differs from the unsupervised approaches in that the scoring function is tuned to a particular relation, e.g., the `locatedAt` relation from Section 2.2.2. In the following, we denote this model as `SkipGram/Supervised`.

2.2.2 Datasets

The following section introduces the datasets that we use for this work. We consider three types of datasets: i) a crowdsourced set of triples expressing the `locatedAt` relation with human judgments, ii) a semi-automatically extracted set of triples expressing the `locatedAt` relation, and iii) a semi-automatically extracted set of `usedFor` triples.

Crowdsourcing of Object-Location Rankings. In order to acquire valid pairs for the `locatedAt` relation we rely on a crowdsourcing approach. In particular, given a certain object, we used crowdsourcing to collect judgments about the likelihood to find this object at a set of predefined locations.

To select the objects and locations for this experiment, every DBpedia entity that falls under the category `Domestic_implements`, or under one of the `narrower` categories than `Domestic_implements` according to SKOS[7], is considered an object. The SPARQL query is given as:

```
select distinct ?object where {
 {
  ?object
    <http://purl.org/dc/terms/subject>
    dbc:Domestic_implements
 } UNION {
  ?object
    <http://purl.org/dc/terms/subject>
```

[6] 5 triples (s', r, o) where we corrupt the subject entity and 5 triples (s, r, o') where the object entity is replaced.

[7] Simple Knowledge Organization System: `https://www.w3.org/2004/02/skos/`

```
    ?category .
  ?category
    <http://www.w3.org/2004/02/skos/core#broader>
    dbc:Domestic_implements .
 }
}
```

Every DBpedia entity that falls under the category `Rooms` is considered a location. The respective query is:

```
 select distinct ?room where {
 ?room
   <http://purl.org/dc/terms/subject>
   dbc:Rooms
}
```

These steps result in 336 objects and 199 locations (as of September 2016). To select suitable pairs expressing the `locatedAt` relation for the creation of the gold standard, we filter out odd or uncommon examples of objects or locations like `Ghodiyu` or `Fainting_room`. We do this by ordering the objects by the number of incoming links to their respective Wikipedia page[8] in descending order and select the 100 top ranking objects for our gold standard. We proceed analogously for the locations, selecting 20 common locations and thus obtain 2,000 object-location pairs in total.

In order to collect the judgments, we set up a crowdsourcing experiment on the CrowdFlower platform[9]. For each of the 2,000 object-location pairs, contributors were asked to rate the likelihood of the object to be in that location on a four-point scale:

- **-2** (**unexpected**): finding the object in the room would cause surprise, e.g., it is unexpected to find a bathtub in a cafeteria.
- **-1** (**unusual**): finding the object in the room would be odd, the object feels out of place, e.g., it is unusual to find a mug in a garage.
- **1** (**plausible**): finding the object in the room would not cause any surprise, it is seen as a normal occurrence, e.g., it is plausible to find a funnel in a dining room.
- **2** (**usual**): the room is the place where the object is typically found, e.g, the kitchen is the usual place to find a spoon.

Contributors were shown ten examples per page, instructions, a short description of the entities (the first sentence from the Wikipedia abstract), a picture (from Wikimedia Commons, when available[10]), and the list of possible answers as labeled radio buttons.

After running the crowdsourcing experiment for a few hours, we collected 12,767 valid judgments, whereas 455 judgments were deemed "untrusted" by CrowdFlower's quality filtering system. The quality control was based on 57 test questions that we provided and a required minimum accuracy of 60% on these questions for a contributor to be considered trustworthy. In total, 440 contributors participated in the experiment.

The pairs received on average 8.59 judgments. Most of the pairs received at least 5 separate judgments, with some outliers collecting more than one hundred judgments each. The average agreement, i.e. the percentage of contributors that answered the most common answer for a given question, is 64.74%. The judgments are skewed towards the negative end of the spectrum, as expected, with 37% pairs rated unexpected, 30% unusual, 24% plausible and 9% usual. The cost of the experiment was 86 USD.

[8] We use the URI counts extracted from the parsing of Wikipedia with the DBpedia Spotlight tool for entity linking [133].

[9] `http://www.crowdflower.com/`

[10] Pictures were available for 94 out of 100 objects.

To use this manually labeled data in later experiments, we normalize, filter and rearrange the scored pairs and obtain three gold standard datasets:

For the first gold standard dataset, we reduce multiple human judgments for each Object-Location pair to a single score by assigning the average of the numeric values. For instance, if the pair (`Wallet, Ballroom`) has been rated -2 (unexpected) six times, -1 (unusual) three times, and never 1 (plausible) or 2 (usual), its score will be about -1.6, indicating that a `Wallet` is not very likely to be found in a `Ballroom`. For each object, we then produce a ranking of all 20 locations by ordering them by their averaged score for the given object. We refer to this dataset of human-labeled rankings as *locatedAt-Human-rankings*.

The second and third gold standard datasets are produced as follows: The contributors' answers are aggregated using relative majority, that is, each object-location pair has exactly one judgment assigned to it, corresponding to the most popular judgment among all the contributors that answered that question. We extract two sets of relations from this dataset to be used as a gold standard for experimental tests: one list of the 156 pairs rated 2 (*usual*) by the majority of contributors, and a larger list of the 496 pairs rated either 1 (*plausible*) or 2 (*usual*). The aggregated judgments in the gold standard have a confidence score assigned to them by CrowdFlower, based on a measure of inter-rater agreement. Pairs that score low on this confidence measure ($\leqslant 0.5$) were filtered out, leaving 118 and 496 pairs, respectively. We refer to these two gold standard sets as *locatedAt-usual* and *locatedAt-usual/plausible*.

Semi-supervised extraction of object-location triples. The SUN database [483] is a large-scale resource for computer vision and object recognition in images. It comprises 131,067 single images, each of them annotated with a label for the type of scene, and labels for each object identified in the scene. The images are annotated with 908 categories based on the type of scene (bedroom, garden, airway, ...). Moreover, 313,884 objects were recognized and annotated with one out of 4,479 category labels.

Despite its original goal of providing high-quality data for training computer vision models, the SUN project generated a wealth of semantic knowledge that is independent from the vision tasks. In particular, the labels are effectively semantic categories of entities such as objects and locations (scenes, using the lexical conventions of the SUN database).

Objects are observed at particular scenes, and this relational information is retained in the database. In total, we extracted 31,407 object-scene pairs from SUN, together with the number of occurrences of each pair. The twenty most occurring pairs are shown in Table 2.5.

According to its documentation, the labels of the SUN database are lemmas from WordNet. However, they are not disambiguated and thus they could refer to any meaning of the lemma. Most importantly for our goals, the labels in their current state are not directly linked to any LOD resource. Faced with the problem of mapping the SUN database completely to a resource like DBpedia, we adopted a safe strategy for the sake of the gold standard creation. We took all the object and scene labels from the SUN pairs for which a resource in DBpedia with matching label exists. In order to limit the noise and obtain a dataset of "typical" location relations, we also removed those pairs that only occur once in the SUN database. This process resulted in 2,961 pairs of entities. We manually checked them and corrected 118 object labels and 44 location labels. In some cases the correct label was already present, so we eliminated the duplicates resulting in a new dataset of 2,935 object-location pairs[11]. The collected triples are used as training data. We refer to this dataset as *locatedAt-Extracted-triples*.

Semi-supervised extraction of object-action triples. While the methods we propose for relation extractions are by design independent of the particular relations they are applied to, we

[11]Of all extracted triples, 24 objects and 12 locations were also among the objects and locations of the crowdsourced dataset.

Table 2.5: Most frequent pairs of object-scene in the SUN database.

Frequency	Object	Scene
1041	wall	b/bedroom
1011	bed	b/bedroom
949	floor	b/bedroom
663	desk_lamp	b/bedroom
650	night_table	b/bedroom
575	ceiling	b/bedroom
566	window	b/bedroom
473	pillow	b/bedroom
463	wall	b/bathroom
460	curtain	b/bedroom
406	painting	b/bedroom
396	floor	b/bathroom
393	cushion	b/bedroom
380	wall	k/kitchen
370	wall	d/dining_room
364	chair	d/dining_room
355	table	d/dining_room
351	floor	d/dining_room
349	cabinet	k/kitchen
344	sky	s/skyscraper

have focused most of our experimental effort towards one kind of relation between objects and locations, namely the typical location where given objects are found. As a first step to assess the generalizability of our approaches to other kinds of relations, we created an alternative dataset revolving around a relation with the same domain as the location relation, i.e., objects, but a very different range, that is, actions. The relation under consideration will be referred to in the rest of the article as `usedFor`, for example the predicate `usedFor(soap, bath)` states that the soap is used for (or, during, in the process of) taking a bath.

We built a dataset of object-action pairs in a `usedFor` relation starting from ConceptNet 5 [282], a large semantic network of automatically collected commonsense facts (see also Section 2.2.5). From the entire ConceptNet, we extracted 46,522 links labeled `usedFor`. Although ConceptNet is partly linked to LOD resources, we found the coverage of such linking to be quite low, especially with respect to non-named entities such as objects. Therefore, we devised a strategy to link as many of the labels involved in `usedFor` relations to DBpedia, without risking to compromise the accuracy of such linking. The strategy is quite simple and it starts from the observation of the data: for the first argument of the relation, we search DBpedia for an entity whose label matches the ConceptNet labels. For the second argument, we search DBpedia for an entity label that matches the gerund form of the ConceptNet label, e.g., *Bath*→*Bathing*. We perform this step because we noticed how actions are usually referred to with nouns in ConceptNet, but with verbs in the gerund form in DBpedia. We used the morphology generation tool for English *morphg* [323] to generate the correct gerund forms also for irregular verbs. The application of this linking strategy resulted in a dataset of 1,674 pairs of DBpedia entities. Table 2.6 shows a few examples of pairs in the dataset.

To use this data as training and test data for the proposed models, we randomly divide the complete set of positive `(Object, usedFor, Action)` triples in a training portion (527 triples) and a test portion (58 triples). We combine each object entity in the test portion with

Table 2.6: Examples of DBpedia entities in a `usedFor` relation, according to ConceptNet and our DBpedia linking strategy.

Object	Action
Machine	Drying
Dictionary	Looking
Ban	Saving
Cake	Jumping
Moon	Lighting
Tourniquet	Saving
Dollar	Saving
Rainbow	Finding
Fast_food_restaurant	Meeting
Clipboard	Keeping

each action entity to generate a complete test set, comprised of positive and negative triples[12]. To account for variations in the performance due to this random partitioning, we repeat each experiment 100 times and report the averaged results in the experimental section. The average size of the test set is ≈ 2059. We refer to this dataset as *usedFor-Extracted-triples*.

2.2.3 Evaluation

This section presents the evaluation of the proposed framework for relation extraction. We apply our models to the data described in Section 2.2.2, consisting of sets of (`Object`, `locatedAt`, `Location`) and (`Object`, `usedFor`, `Action`) triples. These experiments verify the feasibility of our approach for the population of a knowledge base of manipulation relevant data.

We start our experiments by evaluating how well the produced rankings of (`Object`, `locatedAt`, `Location`) triples match the ground truth rankings obtained from human judgments. For this, we i) present the evaluations for the unsupervised methods SkipGram/Cosine and NASARI/-Cosine, ii) show the performance of combinations thereof, and iii) evaluate the newly proposed SkipGram/Supervised method.

The second part of our experiments evaluates how well each proposed method performs in extracting a knowledge base. The evaluation is performed for (`Object`, `locatedAt`, `Location`) and (`Object`, `usedFor`, `Action`) triples.

Ranking evaluation. With the proposed methods from previous sections, we are able to produce a ranking of e.g., locations for a given object that expresses how prototypical the location is for that object. To test the validity of our methods, we compare their output against the gold standard rankings *locatedAt-Human-rankings* that we obtained from the crowdsourced pairs. As a first evaluation, we investigate how well the unsupervised baseline methods perform in creating object-location rankings. Secondly, we show how to improve these results by combining different approaches. Thirdly, we evaluate the supervised model in comparison to our baselines.

Unsupervised object-location ranking evaluation. Apart from the NASARI-based method and the skip-gram-based method we employ two simple baselines for comparison: For the *location frequency* baseline, the object-location pairs are ranked according to the frequency of the

[12]We filter out all generated triples that are falsely labeled as negative in this process.

Table 2.7: Average Precision@k for $k = 1$ and $k = 3$ and average NDCG of the produced rankings against the gold standard rankings.

Method	**NDCG**	**P@1**	**P@3**
Location frequency baseline	.851	.000	.008
Link frequency baseline	.875	.280	.260
NASARI/Cosine	.903	**.390**	.380
SkipGram/Cosine	**.912**	.350	**.400**

location. The ranking is thus the same for each object, since the score of a pair is only computed based on the location. This method makes sense in absence of any further information on the object: e.g., a robot tasked to find an unknown object should inspect "common" rooms such as a kitchen or a studio first, rather than "uncommon" rooms such as a pantry.

The second baseline, the *link frequency*, is based on counting how often every object appears on the Wikipedia page of every location and vice versa. A ranking is produced based on these counts. An issue with this baseline is that the collected counts could be sparse, i.e., most object-location pairs have a count of 0, thus sometimes producing no value for the ranking for an object. This is the case for rather "unusual" objects and locations.

For each object in the dataset, we compare the location ranking produced by our algorithms to the crowdsourced gold standard ranking and compute two metrics: the *Normalized Discounted Cumulative Gain* (NDCG) and the *Precision at k* (Precision@k or P@k).

The NDCG is a measure of rank correlation used in information retrieval that gives more weight to the results at the top of the list than at its bottom. It is defined as follows:

$$NDCG(R) = \frac{DCG(R)}{DCG(R^*)}$$

$$DCG(R) = R_1 + \sum_{i=2}^{|R|} \frac{R_i}{log_2(i+1)}$$

where R is the produced ranking, R_i is the true relevance of the element at position i and R^* is the ideal ranking of the elements in R. R^* can be obtained by sorting the elements by their true relevance scores. This choice of evaluation metric follows from the idea that it is more important to accurately predict which locations are likely for a given object than to decide which are unlikely candidates.

While the NDCG measure gives a complete account of the quality of the produced rankings, it is not easy to interpret apart from comparisons of different outputs. To gain a better insight into our results, we provide an alternative evaluation, the *Precision@k*. The Precision@k measures the number of locations among the first k positions of the produced rankings that are also among the top-k locations in the gold standard ranking. It follows that, with $k = 1$, precision at 1 is 1 if the top returned location is the top location in the gold standard, and 0 otherwise. We compute the average of Precision@k for $k = 1$ and $k = 3$ across all the objects.

Table 2.7 shows the average NDCG and Precision@k across all objects: methods NASARI/Cosine and SkipGram/Cosine, plus the two baselines introduced above.

Both our methods that are based on semantic relatedness outperform the simple baselines with respect to the gold standard rankings. The location frequency baseline performs very poorly, due to an idiosyncrasy in the frequency data, that is, the most "frequent" location in the dataset is *Aisle*. This behavior reflects the difficulty in evaluating this task using only automatic metrics, since automatically extracted scores and rankings may not correspond to common sense judgment.

The NASARI-based similarities outperform the skip-gram-based method when it comes

Table 2.8: Rank correlation and precision at k for the method based on fallback strategy.

Method	NDCG	P@1	P@3
Fallback strategy (threshold=.4)	.907	**.410**	.393
Fallback strategy (threshold=.5)	.906	.400	.393
Fallback strategy (threshold=.6)	.908	**.410**	.406
Fallback strategy (threshold=.7)	.909	.370	.396
Fallback strategy (threshold=.8)	.911	.360	.403
Linear combination (α=.0)	.912	.350	.400
Linear combination (α=.2)	.911	.380	.407
Linear combination (α=.4)	**.913**	.400	**.423**
Linear combination (α=.6)	.911	.390	.417
Linear combination (α=.8)	.910	.390	.410
Linear combination (α=1.0)	.903	.390	.380

to guessing the most likely location for an object (Precision@1), as opposed to the better performance of SkipGram/Cosine in terms of Precision@3 and rank correlation.

We explored the results and found that for 19 objects out of 100, NASARI/Cosine correctly guesses the top ranking location where SkipGram/Cosine fails, while the opposite happens 15 out of 100 times. We also found that the NASARI-based method has a lower coverage than the skip-gram method, due to the coverage of the original resource (NASARI), where not every entity in DBpedia is assigned a vector[13]. The skip-gram-based method also suffers from this problem, however, only for very rare or uncommon objects and locations (as `Triclinium` or `Jamonera`). These findings suggest that the two methods could have different strengths and weaknesses. In the following section we show two strategies to combine them.

Hybrid methods: fallback pipeline and linear combination. The results from the previous sections highlight that the performance of our two main methods may differ qualitatively. In an effort to overcome the coverage issue of NASARI/Cosine, and at the same time experiment with hybrid methods to extract location relations, we devised two simple ways of combining the SkipGram/Cosine and NASARI/Cosine methods. The first method is based on a fallback strategy: given an object, we consider the pair similarity of the object to the top ranking location according to NASARI/Cosine as a measure of confidence. If the top ranked location among the NASARI/Cosine ranking is exceeding a certain threshold, we consider the ranking returned by NASARI/Cosine as reliable. Otherwise, if the similarity is below the threshold, we deem the result unreliable and we adopt the ranking returned by SkipGram/Cosine instead. The second method produces object-location similarity scores by linear combination of the NASARI and skip-gram similarities. The similarity score for the generic pair s, o is thus given by:

$$sim_\alpha(s,o) = \alpha \cdot sim_{NASARI}(s,o) + (1-\alpha) \cdot sim_{SkipGram}(s,o), \tag{2.3}$$

where parameter α controls the weight of one method with respect to the other.

Table 2.8 shows the obtained results, with varying values of the parameters *threshold* and α. While the NDCG is only moderately affected, both Precision@1 and Precision@3 show an increase in performance with Precision@3 showing the highest score of all investigated methods.

Supervised object-location ranking. In the previous experiments, we investigated how well our (unsupervised) baseline methods perform when extracting the `locatedAt` relation. In the following, we compare the earlier results to the performance of a scoring function trained in

[13]Objects like `Backpack` and `Comb`, and locations like `Loft` are all missing.

Table 2.9: Average precision at k for $k = 1$ and $k = 3$ and average NDCG of the produced rankings against the crowdsourced gold standard rankings. *SkipGram/Supervised* denotes the supervised model based on skip-gram embeddings trained for the `locatedAt` relation.

Method	**NDCG**	**P@1**	**P@3**
Location frequency baseline	.851	.000	.008
Link frequency baseline	.875	.280	.260
NASARI/Cosine	.903	.390	.380
SkipGram/Cosine	.912	.350	.400
Linear combination (α=.4)	**.913**	.400	**.423**
SkipGram/Supervised	.908	**.454**	.387

a supervised fashion. For this experiment we train the scoring function in Eq. (2.1) to extract the `locatedAt` relation between objects and locations. The underlying embeddings $\boldsymbol{V}$ on which the scoring function computes its scores are fixed to the skip-gram embeddings $\boldsymbol{V}_{sg}$. We train the supervised method on the semi-automatically extracted triples *locatedAt-Extracted-triples* previously described. These triples act as the positive triples $\mathcal{T}^{+}_{train}$ in the training procedure, from which we also generate the negative examples $\mathcal{T}^{-}_{train}$ following the procedure in Section 2.2.1. We train the model by generating 10 negative triples per positive triple and minimizing the mean squared error from Eq. (2.2). We initialize $\mathbf{M}_r$ with the identity matrix, b_r with 0, and train the model parameter using stochastic gradient descent (SGD) using a learning rate of 0.001. SGD is performed in mini batches of size 100 with 300 epochs of training. The training procedure is realized with *Keras* [112].

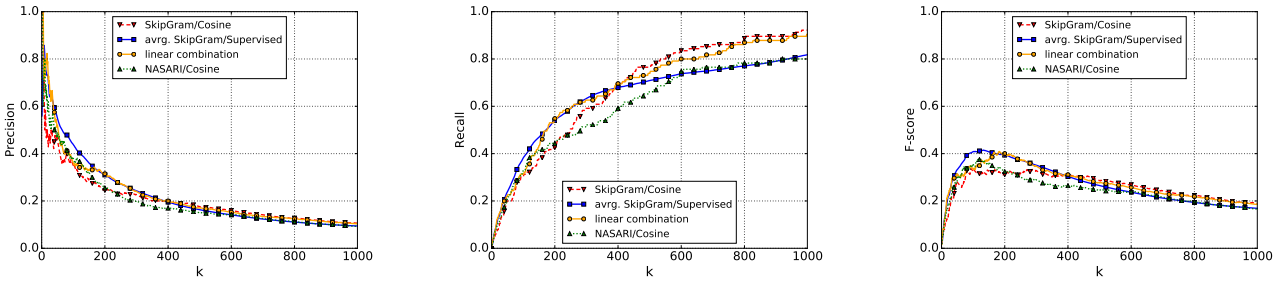
As before, we test the model on the human-rated set of objects and locations *locatedAt-Human-rankings* described in Section 2.2.2 and produce a ranking of locations for each object. Table 2.9 shows the performance of the extended model (SkipGram/Supervised) in comparison to the previous approaches.

Overall, we can observe mixed results. All of our proposed models (supervised and unsupervised) improve upon the baseline methods with respect to all evaluation metrics. Compared to the SkipGram/Cosine model, the SkipGram/Supervised model decreases slightly in performance with respect to the NDCG and more so for the Precision@3 score. Most striking, however, is the increase in Precision@1 of SkipGram/Supervised, showing a relative improvement of 30% to the SkipGram/Cosine model and constituting the highest overall Precision@1 score by a large margin. However, the linear combination (α=.4) still scores higher with respect to Precision@3 and NDCG.

While the presented results do not point to a clear preference for one particular model, in the next section we will investigate the above methods more closely in the context of the generation of a knowledge base.

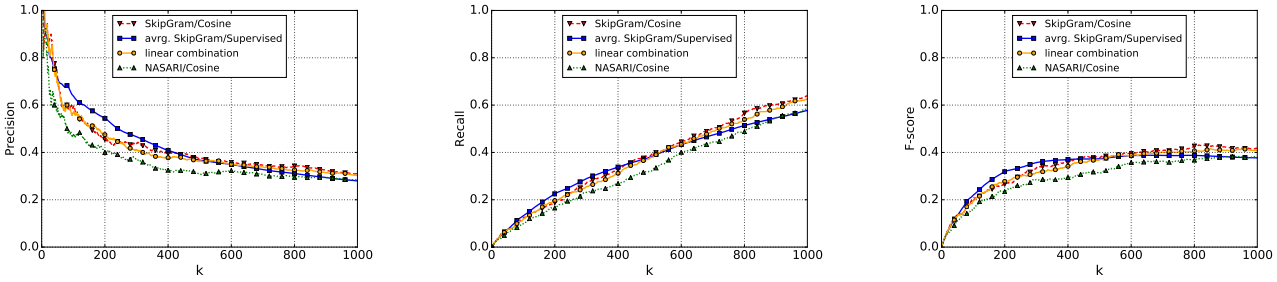
Retrieval evaluation. In the previous section, we tested how the proposed methods perform in determining a ranking of locations given an object. For the purpose of evaluation, the tests have been conducted on a closed set of entities. In this section we return to the original motivation of this work, that is, to collect manipulation-relevant information about objects in an automated fashion in the form of a knowledge base.

All the methods introduced in this work are based on some scoring function of triples expressed as a real number in the range [-1,1] and thus interpretable as a sort of confidence score relative to the target relation. Therefore, by imposing a threshold on the similarity scores and selecting only the object-location pairs that score above said threshold, we can extract a high-confidence set of object-location relations to build a new knowledge base from scratch. Moreover, by using different values for the threshold, we are able to control the quality and the



(a) Precision (b) Recall (c) F-score

Figure 2.9: Evaluation on automatically created knowledge bases ("usual" locations).



(a) Precision (b) Recall (c) F-score

Figure 2.10: Evaluation on automatically created knowledge bases ("plausible" and "usual" locations).

coverage of the produced relations. We test this approach on:

- the *locatedAt-usual* and *locatedAt-usual/plausible* datasets for the `locatedAt` relation between objects and locations, and
- the *usedFor-Extracted-triples* dataset for the `usedFor` relation between objects and actions.

We introduce the `usedFor` relation in order to assess the generalizability of our supervised scoring function.

In general, we extract a knowledge base of triples by scoring each possible candidate triple, thus producing an overall ranking. We then select the top k triples from the ranking, with k being a parameter. This gives us the triples that are considered the most prototypical. We evaluate the retrieved set in terms of Precision, Recall and F-score against the gold standard sets with varying values of k. Here, the precision is the fraction of correctly retrieved triples in the set of all retrieved triples, while the recall is the fraction of retrieved triples that also occur in the gold standard set. The F-score is the harmonic mean of precision and recall:

$$Precision = \frac{|\mathcal{G} \cap \mathcal{R}_k|}{|\mathcal{R}_k|}$$

$$Recall = \frac{|\mathcal{G} \cap \mathcal{R}_k|}{|\mathcal{G}|}$$

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

with $\mathcal{G}$ denoting the set of gold standard triples and $\mathcal{R}_k$ the set of retrieved triples up to rank k.

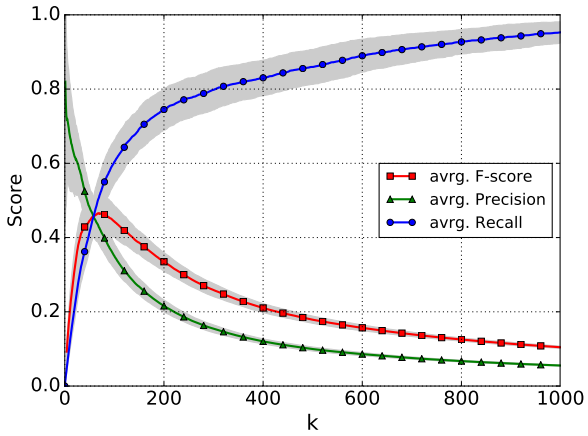


Figure 2.11: Evaluation of knowledge base generation for the `usedFor` relation between objects and actions. Precision, Recall and F-score are given with respect to extracting the to k scored triples.

For the `locatedAt` relation, we also add to the comparison the results of the hybrid, linear combination method, with the best performing parameters in terms of Precision@1, namely the linear combination with $\alpha = 0.4$.

Figures 2.9 and 2.10 show the evaluation of the four methods evaluated against the two aggregated gold standard datasets for the `locatedAt` relation described above. Figures 2.9c and 2.10c, in particular, show F-score plots for a direct comparison of the performance. The SkipGram/Supervised model achieves the highest F-score on the *locatedAt-usual* dataset, peaking at $k = 132$ with an F-score of 0.415. The SkipGram/Cosine model and the linear combination outperform both the NASARI/Cosine and the SkipGram/Supervised in terms of recall, especially for higher k. This also holds for the *locatedAt-usual/plausible* dataset. Here, the SkipGram/Supervised model stands out by achieving high precision values for small values of k. Overall, SkipGram/Supervised performs better for small k (50 – 400) whereas SkipGram/-Cosine and the linear combination obtain better results with increasing k. This seems to be in line with the results from previous experiments in Table 2.9 that show a high Precision@1 for the SkipGram/Supervised model but higher scores for SkipGram/Cosine and the linear combination in terms of Precision@3.

Evaluation of object-action pairs extraction. One of the reasons to introduce a novel technique for relation extraction based on a supervised statistical method, as stated previously, is to be able to scale the extraction across different types of relations. To test the validity of this statement, we apply the same evaluation procedure introduced in the previous part of this section to the `usedFor` relation. For the training and evaluation sets we use the dataset *usedFor-Extracted-triples* comprising of semi-automatically extracted triples from ConceptNet.

Figure 2.11 displays precision, recall and F-score for retrieving the top k results. The results are averaged scores over 100 experiments to account for variations in performance due to the random partitioning in training and evaluation triples and the generation of negative samples. The standard deviation for precision, recall and F-score for all k is visualized along the mean scores.

The supervised model achieves on average a maximum F-score of about 0.465 when extracting 70 triples. This is comparable to the achieved F-scores when training the scoring function for the `locatedAt` relation. To give an insight into the produced false positives, Table 2.10 shows the top 30 extracted triples for the `usedFor` relation of one trained instance of the supervised model.

Table 2.10: A list of the top 30 extracted triples for the `usedFor` relation. The gray highlighted rows mark the entity pairs that are part of the gold standard dataset (Section 2.2.2).

Score	Object	Action
1.00000	Snack	Snacking
0.99896	Snack	Eating
0.99831	Curry	Seasoning
0.99773	Drink	Drinking
0.98675	Garlic	Seasoning
0.98165	Oatmeal	Snacking
0.98120	Food	Eating
0.96440	Pistol	Shooting
0.95218	Drink	Snacking
0.94988	Bagel	Snacking
0.94926	Wheat	Snacking
0.93778	Laser	Printing
0.92760	Food	Snacking
0.91946	Typewriter	Typing
0.91932	Oatmeal	Eating
0.91310	Wok	Cooking
0.89493	Camera	Shooting
0.85415	Coconut	Seasoning
0.85091	Stove	Frying
0.85039	Oatmeal	Seasoning
0.84038	Bagel	Eating
0.83405	Cash	Gambling
0.81985	Oatmeal	Baking
0.80975	Lantern	Lighting
0.80129	Calculator	Typing
0.78279	Laser	Shooting
0.77411	Camera	Recording
0.75712	Book	Writing
0.72924	Stove	Cooking
0.72280	Coconut	Snacking

2.2.4 Building a Knowledge Base of object locations

Given these results, we can aim for a high-confidence knowledge base by selecting the threshold on object-location similarity scores that produces a reasonably high precision knowledge base in the evaluation. For instance, the knowledge base made by the top 50 object-location pairs extracted with the linear combination method ($\alpha = 0.4$) has 0.52 precision and 0.22 recall on the *locatedAt-usual* gold standard (0.70 and 0.07 respectively on the *locatedAt-usual/plausible* set, see Figures 2.9a and 2.10a). The similarity scores in this knowledge base range from 0.570 to 0.866. Following the same methodology that we used to construct the gold standard set of objects and locations (Section 2.2.2), we extract all the 336 `Domestic_implements` and 199 `Rooms` from DBpedia, for a total of 66,864 object-location pairs. Selecting only the pairs whose similarity score is higher than 0.570, according to the linear combination method, yields 931 high confidence location relations. Of these, only 52 were in the gold standard set of pairs (45 were rated "usual" or "plausible" locations), while the remaining 879 are new, such as (`Trivet`, `Kitchen`), (`Flight_bag`, `Airport_lounge`) or (`Soap_dispenser`, `Unisex_public_toilet`). The

distribution of objects across locations has an arithmetic mean of 8.9 objects per location and standard deviation 11.0. `Kitchen` is the most represented location with 89 relations, while 15 out of 107 locations are associated with one single object.[14]

The knowledge base created with this method is the result of one among many possible configurations of a number of methods and parameters. In particular, the creator of a knowledge base involving the extraction of relations is given the choice to prefer precision over recall, or vice-versa. This is done, in our method, by adjusting the threshold on the similarity scores. Employing different algorithms for the computation of the actual similarities (word embeddings vs. entity vectors, supervised vs. unsupervised models) is also expected to result in different knowledge bases. A qualitative assessment of such impact is left for future work.

2.2.5 Related work on mining semantic knowledge from the Web for robotics

To obtain information about unknown objects from the Web, a robot can use perceptual or knowledge-based queries. Future systems will inevitably need to use both. In the previous sections we have focused on the knowledge-based approach, but this can be seen as complementary to systems which use image-based queries to search databases of labeled images for similarity, e.g., [378]. Although the online learning of new *visual* object models is currently a niche area in robotics, some approaches do exist [167, 181]. These approaches are capable of segmenting previously unknown objects in a scene and building models to support their future re-recognition. However, this work focuses purely on visual models (what objects look like), and does not address how the learnt objects are described semantically (what objects are).

The RoboSherlock framework [43] (which we build upon) is one of the most prominent projects to add semantic descriptions to objects for everyday environments, but the framework must largely be configured *a priori* with knowledge of the objects in its environment. It does support more open ended performance, e.g., through the use of Google Goggles, but does not use spatial or semantic context for its Web queries, only vision. The same research group pioneered Web and cloud robotics, where tools such as KnowRob [436] (also used in RoboSherlock) both formalized robot knowledge and capabilities, and used this formal structure to exploit the Web for remote data sources and knowledge sharing. In a more supervised setting, many approaches have used humans to train mobile robots about new objects in their environment [191, 419] and robots have also used Web knowledge sources to improve their performance in closed worlds, e.g., the use of object-room co-occurrence data for room categorization in [211].

The spatial organization of a robot's environment has also been previously exploited to improve task performance. For example, [439, 259] present a system in which the previous experience of spatial arrangements of desktop objects is used to refine the results of a noisy object categorization system. This demonstrates the predictive power of spatial arrangements, which is something we also exploit in our work. However this prior work matched between scenes in the same environment and input modality. In our work we connect spatial arrangements in the robot's situated experience to structured knowledge on the Web. Our predictions for unknown objects rely on determining the semantic relatedness of terms. This is an important topic in several areas, including data mining, information retrieval and web recommendation. [407] applies ontology-based similarity measures in the robotics domain. Background knowledge about all the objects the robot could encounter, is stored in an extended version of the KnowRob ontology. Then, WUP similarity [479] is applied to calculate relatedness of the concept types by considering the depth of the concepts and the depth of their lowest common super-concept in the ontology. [267] presents an approach for computing the semantic relatedness of terms

[14] The full automatically created knowledge base and used resources are available at `https://project.inria.fr/aloof/data/`.

using ontological information extracted from DBpedia for a given domain, using the results for music recommendations. Contrary to these approaches, we compute the semantic relatedness between objects by leveraging the vectorial representation of the DBpedia concepts provided by the NASARI resource [96]. This method links back to earlier distributional semantics work (e.g., Latent Semantic Analysis [262]) with the difference that here concepts are represented as vectors, rather than words.

The work described in Section 2.2 relates to the four research lines discussed below, namely: *i)* machine reading, *ii)* supervised relation extraction, *iii)* encoding common sense knowledge in domain-independent ontologies and knowledge bases, and *iv)* grounding of knowledge from the perspective of cognitive linguistics.

The machine reading paradigm. In the field of knowledge acquisition from the Web, there has been substantial work on extracting taxonomic (e.g., hypernym), part-of relations [194] and complete qualia structures describing an object [115]. Quite recently, there has been a focus on the development of systems that can extract knowledge from any text on any domain (the open information extraction paradigm [166]). The DARPA Machine Reading Program [29] aimed at endowing machines with capabilities for lifelong learning by automatically reading and understanding texts (e.g., [165]). While such approaches are able to quite robustly acquire knowledge from texts, these models are not sufficient to meet our objectives since: *i)* they lack visual and sensorimotor grounding, *ii)* they do not contain extensive object knowledge. While the knowledge extracted by our approach presented here is also not sensorimotorically grounded, we hope that it can support planning of tasks involving object manipulation. Thus, we need to develop additional approaches that can harvest the Web to learn about usages, appearance and functionality of common objects. While there has been some work on grounding symbolic knowledge in language [336], so far there has been no serious effort to compile a large and grounded object knowledge base that can support cognitive systems in understanding objects.

Supervised Relation Extraction. While machine reading attempts to acquire general knowledge by reading texts, other works attempt to extract specific relations using classifiers trained in a supervised approach using labeled data. A training corpus in which the relation of interest is annotated is typically assumed (e.g., [81]). Another possibility is to rely on the so called *distant supervision* approach and use an existing knowledge base to bootstrap the process by relying on triples or facts in the knowledge base to label examples in a corpus (e.g., [225, 226, 225, 432]). Some researchers have modeled relation extraction as a matrix decomposition problem [395]. Other researchers have attempted to train relation extraction approaches in a bootstrapping fashion, relying on knowledge available on the Web, e.g., [58].

Recently, scholars have tried to build models that can learn to extract generic relations from the data, rather than a set of pre-defined relations (see [277] and [63]). Related to these models are techniques to predict triples in knowledge graphs by relying on the embedding of entities (as vectors) and relations (as matrices) in the same distributional space (e.g., TransE [65] and TransH [469]). Similar ideas were tested in computational linguistics in the past years, where relations and modifiers are represented as tensors in the distributional space [31, 138].

Ontologies and KB of common sense knowledge. DBpedia[15] [270] is a large-scale knowledge base automatically extracted from the infoboxes of Wikipedia. Besides its sheer size, it is attractive for the purpose of collecting general knowledge given the one-to-one mapping with Wikipedia (allowing us to exploit the textual and structural information contained in there) and its position as the central hub of the Linked Open Data cloud.

[15] `http://dbpedia.org`

YAGO [431] is an ontology automatically extracted from WordNet and Wikipedia. YAGO extracts facts from the category system and the infoboxes of Wikipedia, and combines these facts with taxonomic relations derived from WordNet. Despite its high coverage, for our goals, YAGO suffers from the same drawbacks as DBpedia, i.e., a lack of knowledge about common objects, that is, about their purpose, functionality, shape, prototypical location, etc.

ConceptNet[16] [282] is a semantic network containing lots of things computers should know about the world. However, we cannot integrate ConceptNet directly in our pipeline because of the low coverage of the mapping with DBpedia— of the 120 DBpedia entities in our gold standard (see Section 2.2.2) only 23 have a correspondent node in ConceptNet.

NELL (Never Ending Language Learning) is the product of a continuously-refined process of knowledge extraction from text [327]. Although NELL is a large-scale and quite fine-grained resource, there are some drawbacks that prevent it to be effectively used as a commonsense knowledge base. The inventory of predicates and relations is very sparse, and categories (including many objects) have no predicates.

OpenCyC[17] [271] attempts to assemble a comprehensive ontology and knowledge base of everyday common sense knowledge, with the goal of enabling AI applications to perform human-like reasoning.

Several projects worldwide have attempted to develop knowledge bases for robots through which knowledge, e.g., about how to manipulate certain objects, can be shared among many robots. Examples of such platforms are the RoboEarth project [462], RoboBrain [405] or KnowRob [435], discussed before.

While the above resources are without doubt very useful, we are interested in developing an approach that can extract new knowledge leveraging text corpora, complementing the knowledge contained in ontologies and knowledge bases such as the ones described above.

Grounded Knowledge and Cognitive Linguistics Many scholars have argued that, from a cognitive perspective, knowledge needs to be grounded [213] as well as modality-specific to support simulation, a mental activity that is regarded as ubiquitous in cognitive intelligent systems [34]. Other seminal work has argued that cognition is categorical [214, 215] and that perceptual and cognitive reasoning rely on schematic knowledge. In particular, there has been substantial work on describing the schemas by which we perceive and understand spatial knowledge [434]. The knowledge we have gathered is neither grounded nor schematic, nor modality-specific in the above senses, but rather amodal and symbolic. This type of knowledge is arguably useful in high-level planning but clearly is not sufficient to support simulation or event action execution. Developing models by which natural language can be grounded in action has been the concern of other authors, e.g., Misra et al. [325] as well as Bollini et al. [64]. Some work has considered extracting spatial relations in natural language input [253]. Differently from the above mentioned works, we are neither interested in interpreting natural language with respect to grounded action representations nor in extracting spatial relations from a given sentence. Rather, our goal is to extract prototypical common sense background knowledge from large corpora.

2.3 Natural Language Processing of Song Lyrics

Given that lyrics encode an important part of the semantics of a song, this Section is dedicated to the description of my research activity on extracting relevant information from the lyrics, such as their structure segmentation, their topics, the explicitness of the lyrics content, the

[16] `http://conceptnet5.media.mit.edu/`

[17] `http://www.opencyc.org/` as RDF representations: `http://sw.opencyc.org/`

salient passages of a song and the emotions conveyed. This research was done in the context of the WASABI project.

Understanding the structure of song lyrics (e.g., intro, verse, chorus) is an important task for music content analysis since it allows to split a song into semantically meaningful segments enabling a description of each section rather than a global description of the whole song. Carrying out this task by means of an automated system is a challenging but useful task, that would allow to enrich song lyrics with improved structural clues that can be used for instance by search engines handling real-word large song collections.

2.4 Lyrics Segmentation via bimodal text-audio representation

Understanding the structure of song lyrics (e.g., intro, verse, chorus) is an important task for music content analysis [106, 473] since it allows to split a song into semantically meaningful segments enabling a description of each section rather than a global description of the whole song. The importance of this task arises also in Music Information Retrieval, where music structure detection is a research area aiming at automatically estimating the temporal structure of a music track by analyzing the characteristics of its audio signal over time. Given that lyrics contain rich information about the semantic structure of a song, relying on textual features could help in overcoming the existing difficulties associated with large acoustic variation in music. However, so far only a few works have addressed the task lyrics-wise [175, 296, 473, 27]. Carrying out structure detection by means of an automated system is therefore a challenging but useful task, that would allow to enrich song lyrics with improved structural clues that can be used for instance by search engines handling real-word large song collections. A step forward, a complete music search engine should support search criteria exploiting both the audio and the textual dimensions of a song.

Structure detection consists of two steps: a text segmentation stage that divides lyrics into segments, and a semantic labelling stage that labels each segment with a structure type (e.g., intro, verse, chorus). Given the variability in the set of structure types provided in the literature according to different genres [433, 71], rare attempts have been made to achieve the second step, i.e. semantic labelling. While addressing the first step is the core contribution of this section, we leave the task of semantic labelling for future work.

In [175] we proposed a first neural approach for lyrics segmentation that was relying on purely textual features. However, with this approach we fail to capture the structure of the song in case there is no clear structure in the lyrics - when sentences are never repeated or in the opposite case when they are always repeated. In such cases however, the structure may arise from the acoustic/audio content of the song, often from the melody representation. This section aims at extending the approach proposed in [175] by complementing the textual analysis with acoustic aspects. We perform lyrics segmentation on a synchronized text-audio representation of a song to benefit from both textual and audio features.

In this direction, this work focuses on the following research question: *given the text and audio of a song, can we learn to detect the lines delimiting segments in the song text?* This question is broken down into two sub questions: *1) given solely the song text, can we learn to detect the lines delimiting segments in the song?* and *2) do audio features - in addition to the text - boost the model performance on the lyrics segmentation task?*

To address these questions, this article contains the following contributions:

1. We introduce a convolutional neural network-based model that *i)* efficiently exploits the Self-Similarity Matrix representations (SSM) used in the state-of-the-art [473], and *ii)*

can utilize traditional features alongside the SSMs.

2. We experiment with novel features that aim at revealing different properties of a song text, such as its phonetics and syntax. We evaluate this **unimodal** (purely text-based) approach on two standard datasets of English lyrics, the Music Lyrics Database and the WASABI corpus. We show that our proposed method can effectively detect the boundaries of music segments outperforming the state of the art, and is portable across collections of song lyrics of heterogeneous musical genre.

3. We experiment with a **bimodal** lyrics representation that incorporates audio features into our model. For this, we use a novel bimodal corpus (DALI) in which each song text is time-aligned to its associated audio. Our bimodal lyrics segmentation performs significantly better than the unimodal approach. We investigate which text and audio features are the most relevant to detect lyrics segments and show that the text and audio modalities complement each other. We perform an ablation test to find out to what extent our method relies on the alignment quality of the lyrics-audio segment representations.

To better understand the rational underlying the proposed approach, consider the segmentation of the Pop song depicted in Figure 2.12. The left side shows the lyrics and its segmentation into its structural parts: the horizontal green lines indicate the segment borders between the different lyrics segments. We can summarize the segmentation as follows: Verse_1-Verse_2-Bridge_1-Chorus_1-Verse_3-Bridge_2-Chorus_2-Chorus_3-Chorus_4-Outro. The middle of Figure 2.12 shows the repetitive structure of the lyrics. The exact nature of this structure representation is introduced later and is not needed to understand this introductory example. The crucial point is that the segment borders in the song text (green lines) coincide with highlighted rectangles in the chorus (the C_i) of the lyrics structure (middle). We find that in the verses (the V_i) and bridges (the B_i) highlighted rectangles are only found in the melody structure (right). The reason is that these verses have different lyrics, but share the same melody (analogous for the bridges). While the repetitive structure of the lyrics is an effective representation for lyrics segmentation, we believe that an enriched segment representation that also takes into account the audio of a song can improve segmentation models. While previous approaches relied on purely textual features for lyrics segmentation, showing the discussed limitations, we propose to perform lyrics segmentation on a synchronized text-audio representation of a song to benefit from both textual and audio features.

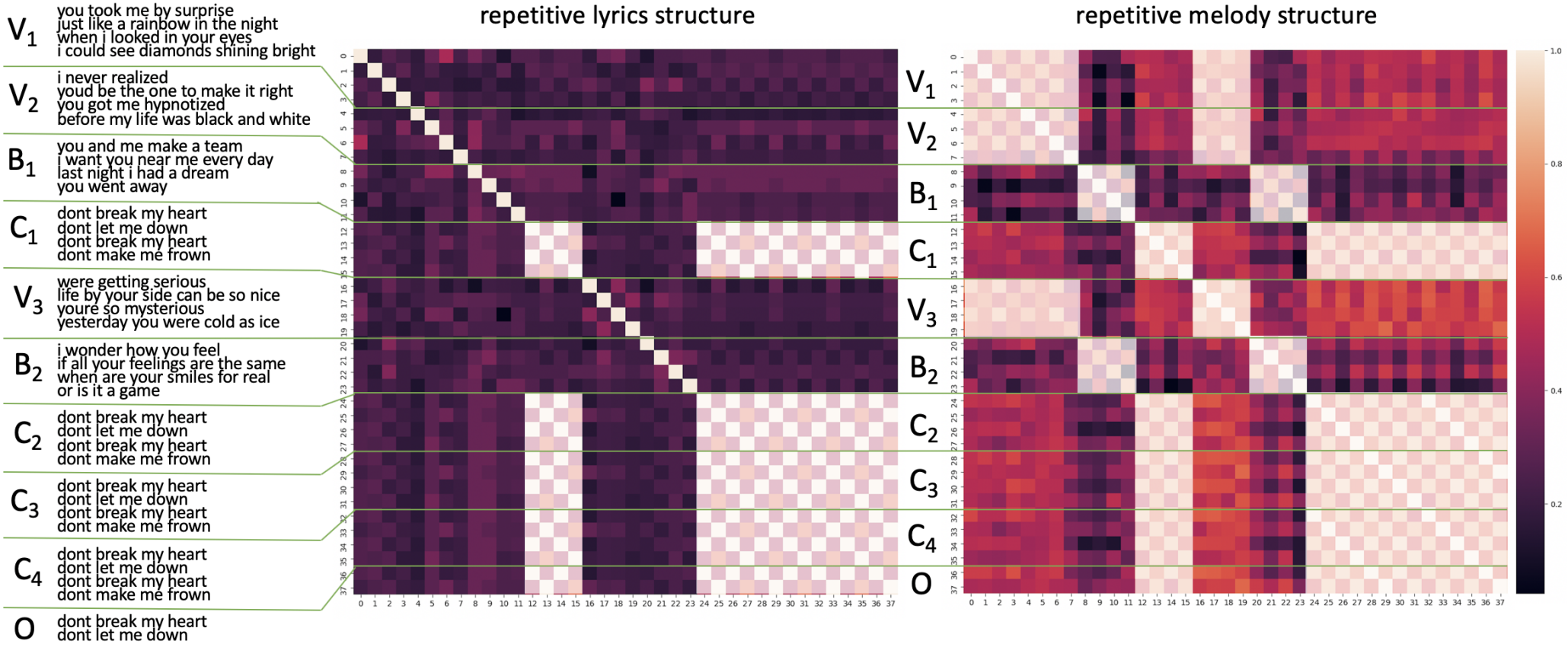


Figure 2.12: Lyrics (left) of a Pop song, the repetitive structure of the lyrics (middle), and the repetitive structure of the song melody (right). Lyrics segment borders (green lines) coincide with highlighted rectangles in lyrics structure and melody structure. ("Don't Break My Heart" by Den Harrow)

2.4.1 Modelling segments in song lyrics

Detecting the structure of a song text is a non-trivial task that requires diverse knowledge and consists of two steps: text segmentation followed by segment labelling. In this work we focus on the task of segmenting the lyrics. This first step is fundamental to segment labelling when segment borders are not known. Even when segment borders are "indicated" by line breaks in lyrics available online, those line breaks have usually been annotated by users and neither are they necessarily identical to those intended by the songwriter, nor do users in general agree on where to put them. Thus, a method to automatically segment unsegmented song texts is needed to automate that first step. Many heuristics can be imagined to find the segment borders. In our example, separating the lyrics into segments of a constant length of four lines (Figure 2.12) gives the correct segmentation. However, in another example, the segments can be of different length. This is to say that enumerating heuristic rules is an open-ended task.

We follow [473] by casting the lyrics segmentation task as binary classification. Let $L = \{a_1, a_2, ..., a_n\}$ be the lyrics of a song composed of n lyrics lines and $seg \subseteq (L, \mathbb{B})$ be a function that returns for each line $a_i \in L$ if it is the end of a segment. The task is to learn a classifier that approximates seg. At the learning stage, the ground truth segment borders are observed from segmented text as double line breaks. At the testing stage the classifier has to predict the now hidden segment borders.

Note that lyrics lines do not exist in isolation, as lyrics are texts that accompany music. Therefore, each lyrics line is naturally associated to a segment of audio. We define a bimodal lyrics line $a_i = (l_i, s_i)$ as a pair containing both the i-th text line l_i, and its associated audio segment s_i. In the case we only use the textual information, we model this as unimodal lyrics lines, i.e. $a_i = (l_i)$.[18]

In order to infer the lyrics structure, we rely on our Convolutional Neural Network-based model that we introduced in [175]. Our model architecture is detailed later in this section. It detects segment boundaries by leveraging the repeated patterns in a song text that are conveyed by the Self-Similarity Matrices.

Self-Similarity Matrices. We produce Self-Similarity Matrices (SSMs) based on bimodal lyrics lines $a_i = (l_i, s_i)$ in order to capture repeated patterns in the text line l_i as well as its associated audio segment s_i. SSMs have been previously used in the literature to estimate the structure of music [183, 118] and lyrics [473, 175]. Given a song consisting of bimodal lines $\{a_1, a_2, ..., a_n\}$, a Self-Similarity Matrix $SSM_M \in \mathbb{R}^{n \times n}$ is constructed, where each element is set by computing a similarity measure between the two corresponding elements $(SSM_{\mathrm{M}})_{ij} = \mathrm{sim}_{\mathrm{M}}(x_i, x_j)$. We choose x_i, x_j to be elements from the same modality, i.e. they are either both lyrics lines (l_i) or both audio segments (s_i) associated to lyrics lines. $\mathrm{sim}_{\mathrm{M}}$ is a similarity measures that compares two elements of the same modality to each other. In our experiments, this is either a text-based or an audio-based similarity. As a result, SSMs constructed from a text-based similarity highlight distinct patterns of the text, revealing the underlying structure (see Figure 2.12, middle). Analogously, SSMs constructed from an audio-based similarity highlight distinct patterns of the audio (see Figure 2.12, right). In the unimodal case, we compute SSMs from only one modality: either text lines l_i or audio segments s_i.

There are two common patterns that were investigated in the literature: diagonals and rectangles. Diagonals parallel to the main diagonal indicate sequences that repeat and are typically found in a chorus. Rectangles, on the other hand, indicate sequences in which all the lines are highly similar to one another. Both of these patterns were found to be indicators of segment borders.

[18] This definition can be straightforwardly extended to more modalities, a_i then becomes a tuple containing time-synchronized information.

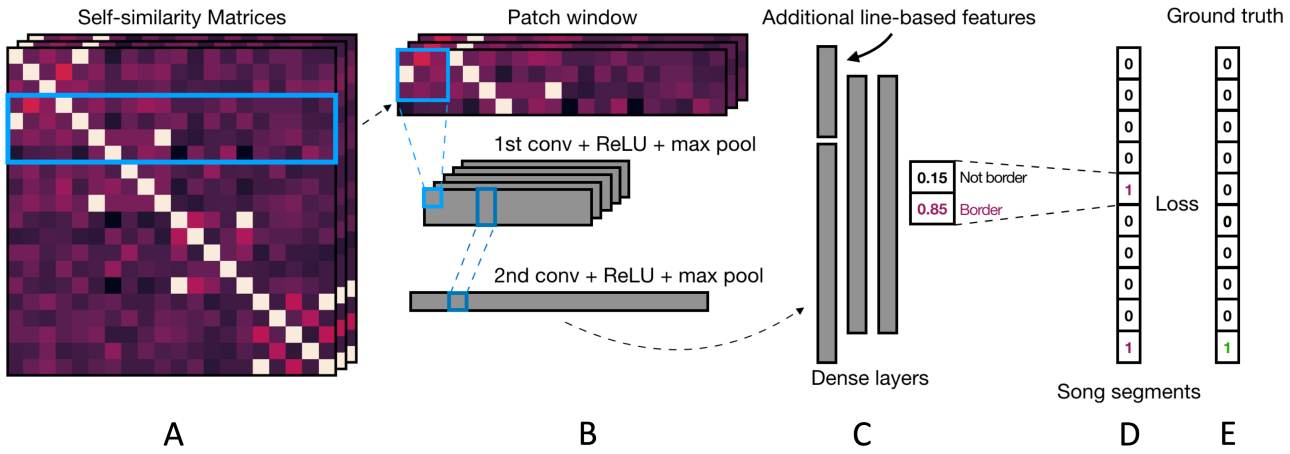


Figure 2.13: Convolutional Neural Network-based model inferring lyrics segmentation.

Convolutional Neural Network-based Model. Lyrics segments manifest themselves in the form of distinct patterns in the SSM. In order to detect these patterns efficiently, we introduce the Convolutional Neural Network (CNN) architecture which is illustrated in Figure 2.13. The model predicts for each lyrics line if it is segment ending. For each of the n lines of a song text the model receives patches (see Figure 2.13, step A) extracted from SSMs $\in \mathbb{R}^{n \times n}$ and centered around the line: $\text{input}_i = \{P_i^1, P_i^2, ..., P_i^c\} \in \mathbb{R}^{2w \times n \times c}$, where c is the number of SSMs or number of channels and w is the window size. To ensure the model captures the segment-indicating patterns regardless of their location and relative size, the input patches go through two convolutional layers (see Figure 2.13, step B) [195], using filter sizes of $(w+1) \times (w+1)$ and $1 \times w$, respectively. By applying max pooling after both convolutions each feature is down-sampled to a scalar. After the convolutions, the resulting feature vector is concatenated with the line-based features (see Figure 2.13, step C) and goes through a series of densely connected layers. Finally, the $softmax$ is applied to produce probabilities for each class (border/not border) (see Figure 2.13, step D). The model is trained with supervision using binary cross-entropy loss between predicted and ground truth segment border labels (see Figure 2.13, step E). Note that while the patch extraction is a local process, the SSM representation captures global relationships, namely the similarity of a line to all other lines in the lyrics.

Bimodal Lyrics Lines. To perform lyrics segmentation on a bimodal text-audio representation of a song to benefit from both textual and audio features, we use a corpus where the annotated lyrics ground truth (segment borders) is synchronized with the audio. This bimodal dataset is described in the following subsection. We focus solely on the audio extracts that have singing voice, as only they are associated to the lyrics. For that let t_i be the time interval of the (singing event of) text line l_i in our synchronized text-audio corpus. Then, a bimodal lyrics line $a_i = (l_i, s_i)$ consists of both a text line l_i (the text line during t_i) and its associated audio segment s_i (the audio segment during t_i). As a result, we have the same number of text lines and audio segments. While the textual information l_i can be used directly to produce SSMs, the complexity of the raw audio signal prevents it from being used as direct input of our system. Instead, it is common to extract features from the audio that highlight some aspects of the signal that are correlated with the different musical dimensions. Therefore, we describe each audio segment s_i as set of different time vectors. Each frame of a vector contains information of a precise and small time interval. The size of each audio frame depends on the configuration of each audio feature. We call an audio segment s_i *featurized* by a feature f if f is applied to all frames of s_i. For our bimodal segment representation we featurize each s_i with one of the following features:

- **Mel-frequency cepstral coefficients ($\boldsymbol{mfcc} \in \mathbb{R}^{14}$):** these coefficients [137] emphasize parts of the signal that are related with our understanding of the musical timbre. They have proven to be very efficient in a large range of audio applications.

- **Chroma feature ($\boldsymbol{chr} \in \mathbb{R}^{12}$):** this feature [188] describes the harmonic information of each frame by computing the "presence" of the twelve different notes.

2.4.2 Datasets

Song texts are available widely across the Web in the form of user-generated content. Unfortunately for research purposes, there is no comprehensive publicly available online resource that would allow a more standardized evaluation of research results. This is mostly attributable to copyright limitations and has been criticized before in [301]. Research therefore is usually undertaken on corpora that were created using standard web-crawling techniques by the respective researchers. Due to the user-generated nature of song texts on the Web, such crawled data is potentially noisy and heterogeneous, e.g., the way in which line repetitions are annotated can range from verbatim duplication to something like *Chorus (4x)* to indicate repeating the chorus four times.

In the following we describe the lyrics corpora we used in our experiments. First, MLDB and WASABI are purely textual corpora. Complementarily, DALI is a corpus that contains bimodal lyrics representations in which text and audio are synchronized.

MLDB and WASABI. The Music Lyrics Database (MLDB) V.1.2.7[19] is a proprietary lyrics corpus of popular songs of diverse genres. We use this corpus in the same configuration as used before by the state of the art in order to facilitate a comparison with their work. Consequently, we only consider English song texts that have five or more segments and we use the same training, development and test indices, which is a 60%-20%-20% split. In total we have 103k song texts with at least 5 segments. 92% of the remaining song texts count between 6 and 12 segments.

The WASABI corpus[20] [317], is a larger corpus of song texts, consisting of 744k English song texts with at least 5 segments, and for each song it provides the following information: its lyrics[21], the synchronized lyrics when available[22], DBpedia abstracts and categories the song belongs to, genre, label, writer, release date, awards, producers, artist and/or band members, the stereo audio track from Deezer, when available, the unmixed audio tracks of the song, its ISRC, bpm, and duration.

DALI. The DALI corpus[23] [316] contains synchronized lyrics-audio representations on different levels of granularity: syllables, words, lines and segments / paragraphs. It was created by joining two datasets: (1) a corpus for karaoke singing (AMX) which contains alignments between lyrics and audio on the syllable level and (2) a subset of WASABI lyrics that belong to the same songs than the lyrics in AMX. Note that corresponding lyrics in WASABI can differ from those in AMX to some extent. Also, in AMX there is no annotation of segments. DALI provides estimated segments for AMX lyrics, projected from the ground truth segments from WASABI. For example, Figure 2.14 shows on the left side the lyrics lines as given in AMX. The right side shows the lyrics lines given in WASABI as well as the ground truth lyrics segments.

[19] `http://www.odditysoftware.com/page-datasales1.htm`
[20] `https://wasabi.i3s.unice.fr/`
[21] Extracted from `http://lyrics.wikia.com/`
[22] From `http://usdb.animux.de`
[23] https://github.com/gabolsgabs/DALI

Corpus name	Alignment quality	Song count
Q^+	high (90-100%)	1048
Q^0	med (52-90%)	1868
Q^-	low (0-52%)	1868
full dataset	-	4784

Table 2.11: The DALI dataset partitioned by alignment quality

The left side shows the estimated lyrics segments in AMX. Note how the lyrics in WASABI have one segment more, as the segment W_3 has no counter part in AMX.

Based on the requirements for our task, we derive a measure to assess how well the estimated AMX segments correspond / align to the groundtruth WASABI segments. Since we will use the WASABI segments as ground truth labels for supervised learning, we need to make sure, the AMX lines (and hence audio information) actually belongs to the aligned segment. As only for the AMX lyrics segments we have aligned audio features and we want to consistently use audio features in our segment representations, we make sure that every AMX segment has a counterpart WASABI segment (see Figure 2.14, $A_0 \sim W_0$, $A_1 \sim W_1$, $A_2 \sim W_2$, $A_3 \sim W_4$). On the other hand, we allow WASABI segments to have no corresponding AMX segments (see Figure 2.14, W_3). We further do not impose constraints on the order of appearance of segments in AMX segmentations vs. WASABI segmentations, to allow for possible rearrangements in the order of corresponding segments. With these considerations, we formulate a measure of alignment quality that is tailored to our task of bimodal lyrics segmentation. Let A, W be segmentations, where $A = A_0A_1...A_x$ and the A_i are AMX segments and $W = W_0W_1...W_y$ with WASABI lyrics segments W_i. Then the alignment quality between the segmentations A, W is composed from the similarities of the best-matching segments. Using string similarity sim_{str} as defined in the following subsection, we define the alignment quality $Qual$ as follows:

$$\begin{aligned} Qual(A, W) &= Qual(A_0A_1...A_x,\ W_0W_1...W_y) \\ &= \min_{0 \le i \le x} \{\ \max_{0 \le j \le y} \{\ \text{sim}_{\text{str}}(A_i, W_j)\ \}\ \} \end{aligned}$$

In order to test the impact of $Qual$ on the performance of our lyrics segmentation algorithm, we partition the DALI corpus into parts with different $Qual$. Initially, DALI consists of 5358 lyrics that are synchronized to their audio track. Like in previous publications [473, 175], we ensure that all song texts contain at least 5 segments. This constraint reduces the number of tracks used by us to 4784. We partition the 4784 tracks based on their $Qual$ into high (Q^+), med (Q^0), and low (Q^-) alignment quality datasets. Table 5.1 gives an overview over the resulting dataset partitions. The Q^+ dataset consists of 50842 lines and 7985 segment borders and has the following language distribution: 72% English, 11% German, 4% French, 3% Spanish, 3% Dutch, 7% other languages.

2.4.3 Experimental setting

The central input features used by our Convolutional Neural Network-based model are the different SSMs. Therefore, the choice of similarities used to produce the SSMs is essential to the approach. We experiment with both text-based and audio-based similarities, described in the following paragraph. We then detail the model configurations and hyperparameters. We describe separately the settings for the experiments using unimodal lyrics lines (text only) or bimodal lyrics lines (text and audio), respectively.

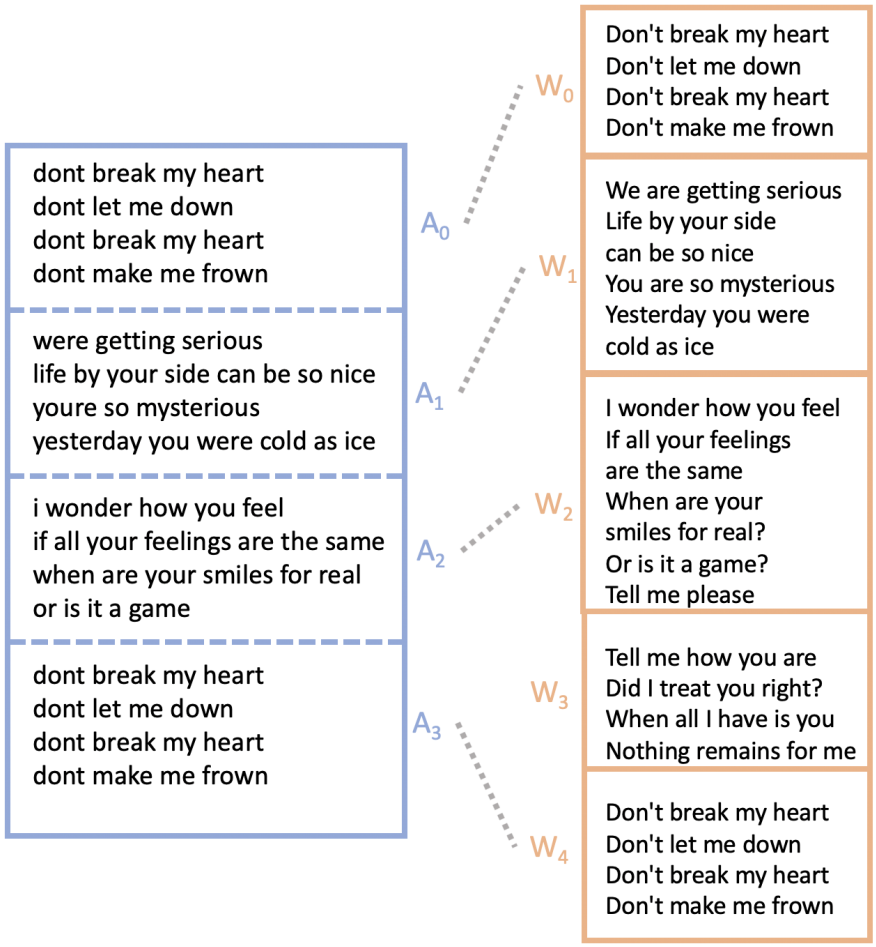


Figure 2.14: Lyrics lines and estimated lyrics segments in AMX (left). Lyrics lines and ground truth lyrics segments in WASABI (right) for the song ("Don't Break My Heart" by Den Harrow)

Similarity Measures. We produce SSMs based on three line-based text similarity measures. We further add audio-based similarities - the crucial ingredient that makes our approach multimodal. In the following, we define the text-based and audio-based similarities used to compute the SSMs.

Text similarities: given the text lines of the lyrics, we compute different similarity measures, based on either their characters, their phonetics or their syntax.

- **String similarity ($\mathbf{sim_{str}}$):** a normalized Levenshtein string edit similarity between the characters of two lines of text [272]. This has been widely used - e.g., [473, 175].
- **Phonetic similarity ($\mathbf{sim_{phon}}$):** a simplified phonetic representation of the lines computed using the "Double Metaphone Search Algorithm" [379]. When applied to "i love you very much" and "i'l off you vary match" it returns the same result: "ALFFRMX". This algorithm was developed to capture the similarity of similar sounding words even with possibly very dissimilar orthography. We translate the text lines into this "phonetic language" and then compute sim_{str} between them.
- **Lexico-syntactical similarity ($\mathbf{sim_{lsyn}}$):** this measure, initially proposed in [170], combines lexical with syntactical similarity. sim_{lex} captures the similarity between text lines such as "Look into my eyes" and "I look into your eyes": these are partially similar on a lexical level and partially similar on a syntactical level. Given two lines x, y lexico-

syntactical similarity is defined as: $sim_{lsyn}(x,y) = sim_{lex}^2(x,y) + (1 - sim_{syn}) \cdot sim_{syn}(\hat{x},\hat{y})$, where sim_{lex} is the overlap of the bigrams of words in x and y, and sim_{syn} is the overlap of the bigrams of pos tags in $\hat{x},\hat{y}$, the remaining tokens that did not overlap on a word level.

Audio similarities: There are several alternatives to measure the similarity between two audio sequences (e.g., mfcc sequences) of possibly different lengths, among which Dynamic Time Warping T_d is the most popular one in the Music Information Retrieval community. Given bimodal lyrics lines a_u, a_v (as previously defined), we compare two audio segments s_u and s_v that are featurized by a particular audio feature (mfcc, chr) using T_d:

$$T_d(i,j) = d(s_u(i), s_v(j)) + \min \left\{ \begin{array}{c} T_d(i-1,j), \\ T_d(i-1,j-1), \\ T_d(i,j-1) \end{array} \right\}$$

T_d must be parametrized by an inner distance d to measure the distance between the frame i of s_u and the frame j of s_v. Depending on the particular audio feature s_u and s_v are featurized with, we employ a different inner distance as defined below. Let m be the length of the vector s_u and n be the length of s_v. Then, we compute the minimal distance between the two audio sequences as $T_d(m,n)$ and normalize this by the length r of the shortest alignment path between s_u and s_v to obtain values in [0,1] that are comparable to each other. We finally apply $\lambda x.(1-x)$ to turn the distance T_d into a similarity measure S_d:

$$S_d(s_u, s_v) = 1 - T_d(m,n) \cdot r^{-1}$$

Given bimodal lyrics lines a_i, we now define similarity measures between audio segments s_i that are featurized by a particular audio feature presented previously (mfcc, chr) based on our similarity measure S_d:

- **MFCC similarity ($\mathbf{sim_{mfcc}}$):** S_d between two audio segments featurized by the mfcc feature. As inner distance we use the cosine distance: $d(x,y) = x \cdot y \cdot (\|x\| \cdot \|y\|)^{-1}$
- **Chroma similarity ($\mathbf{sim_{chr}}$):** S_d between two audio segments featurized by the chroma feature; using cosine distance as inner distance

Models and configurations. In our first experiment we represent song texts via **unimodal lyrics lines** and experiment **on the MLDB and WASABI datasets**. We compare to the state of the art [473] and successfully reproduce their best features to validate their approach. Two groups of features are used in the replication: repeated pattern features (RPF) extracted from SSMs and n-grams extracted from text lines. The RPF basically act as hand-crafted image filters that aim to detect the edges and the insides of diagonals and rectangles in the SSM.

Then, our own models are neural networks, that use as features SSMs and two line-based features: the line length and n-grams. For the line length, we extracted the character count from each line, a simple proxy of the orthographic shape of the song text. Intuitively, segments that belong together tend to have similar shapes. Similarly to [473]'s term features we extracted those n-grams from each line that are most indicative for segment borders: using the tf-idf weighting scheme, we extracted n-grams that are typically found left or right from the segment border, varied n-gram lengths and also included indicative part-of-speech tag n-grams. This resulted in 240 term features in total. The most indicative words at the start of a segment were: {ok, lately, okay, yo, excuse, dear, well, hey}. As segment-initial phrases we found: {Been a long, I've been, There's a, Won't you, Na na na, Hey, hey}. Typical words ending a segment

were: {..., .., !, ., yeah, ohh, woah. c'mon, wonderland}. And as segment-final phrases we found as most indicative: {yeah!, come on!, love you., !!!, to you., with you., check it out, at all., let's go, ...}

In this experiment we consider only SSMs made from text-based similarities; we note this in the model name as CNN_{text}. We further name a CNN model by the set of SSMs that it uses as features. For example, the model $\text{CNN}_{\text{text}}\{\text{str}\}$ uses as only feature the SSM made from string similarity sim_{str}, while the model $\text{CNN}_{\text{text}}\{\text{str, phon, lsyn}\}$ uses three SSMs in parallel (as different input channels), one from each similarity.

For convolutional layers we empirically set $w_{\text{size}} = 2$ and the amount of features extracted after each convolution to 128. Dense layers have 512 hidden units. We have also tuned the learning rate (negative degrees of 10), the dropout probability with increments of 0.1. The batch size was selected from the beginning to be 256 to better saturate our GPU. The CNN models were implemented using Tensorflow.

For comparison, we implement two baselines. The random baseline guesses for each line independently if it is a segment border (with a probability of 50%) or not. The line length baseline uses as only feature the line length in characters and is trained using a logistic regression classifier.

For comparison with the state of the art, we use as a first dataset the same they used, the MLDB (see Section 2.4.2). To test the system portability to bigger and more heterogeneous data sources, we further experimented our method on the WASABI corpus (see Section 2.4.2). In order to test the influence of genre on classification performance, we aligned MLDB to WASABI as the latter provides genre information. Song texts that had the exact same title and artist names (ignoring case) in both data sets were aligned. This rather strict filter resulted in an amount of 58567 (57%) song texts with genre information in MLDB. Table 2.13 shows the distribution of the genres in MLDB song texts. We then tested our method on each genre separately, to test our hypothesis that classification is harder for some genres in which almost no repeated patterns can be detected (as Rap songs). To the best of our knowledge, previous work did not report on genre-specific results.

In this work we did not normalize the lyrics in order to rigorously compare our results to [473]. We estimate the proportion of lyrics containing tags such as *Chorus* to be marginal (0.1-0.5%) in the MLDB corpus. When applying our methods for lyrics segmentation to lyrics found online, an appropriate normalization method should be applied as a pre-processing step. For details on such a normalization procedure we refer the reader to [170], Section 2.1.

Evaluation metrics are Precision (P), Recall (R), and f-score (F_1). Significance is tested with a permutation test [351], and the p-value is reported.

Then, in the second experiment, the song texts are represented via **bimodal lyrics lines** and experimentation is performed **on the DALI corpus**. In order to test our hypotheses which text and audio features are most relevant to detect segment boundaries, and whether the text and audio modalities complement each other, we compare different types of models: baselines, text-based models, audio-based models, and finally bimodal models that use both text and audio features. We provide the following baselines: the random baseline guesses for each line independently if it is a segment border (with a probability of 50%) or not. The line length baselines use as feature only the line length in characters (text-based model) or milliseconds (audio-based model) or both, respectively. These baselines are trained using a logistic regression classifier. All other models are CNNs using the architecture described previously and use as features SSMs made from different textual or audio similarities as previously described. The CNN-based models that use purely textual features (str) are named CNN_{text}, while the CNN-based models using purely audio features (mfcc, chr) are named $\text{CNN}_{\text{audio}}$. Lastly, the CNN_{mult} models are multimodal in the sense that they use combinations of textual and audio features.

We name a CNN model by its modality (text, audio, mult) as well as by the set of SSMs that it uses as features. For example, the model $\text{CNN}_{\text{mult}}\{\text{str, mfcc}\}$ uses as textual feature the SSM made from string similarity sim_{str} and as audio feature the SSM made from mfcc similarity sim_{mfcc}.

As dataset we use the Q^+ part of the DALI dataset. We split the data randomly into training and test sets using the following scheme: considering that the DALI dataset is relatively small, we average over two different 5-fold cross-validations. We prefer this sampling strategy for our small dataset over a more common 10-fold cross-validation as it avoids the test set becoming too small.

2.4.4 Results and discussion

Table 2.12 shows the **results of our experiments with unimodal lyrics lines on the MLDB dataset.** We start by measuring the performance of our replication of [473]'s approach. This reimplementation exhibits 56.3% F_1, similar to the results reported in the original paper (57.7%). The divergence could be attributed to a different choice of hyperparameters and feature extraction code. Much weaker baselines were explored as well. The random baseline resulted in 18.6% F_1, while the usage of simple line-based features, such as the line length (character count), improves this to 25.4%.

The best CNN-based model, $\text{CNN}_{\text{text}}\{\text{str, phon, lsyn}\} + \text{n-grams}$, outperforms all our baselines reaching 67.4% F_1, 8.2pp better than the results reported in [473]. We perform a permutation test [351] of this model against all other models. In every case, the performance difference is statistically significant ($p < .05$).

Subsequent feature analysis revealed that the model $\text{CNN}_{\text{text}}\{\text{str}\}$ is by far the most effective. The $\text{CNN}_{\text{text}}\{\text{lsyn}\}$ model exhibits much lower performance, despite using a much more complex feature. We believe the lexico-syntactical similarity is much noisier as it relies on n-grams and PoS tags, and thus propagates error from the tokenizers and PoS taggers. The $\text{CNN}_{\text{text}}\{\text{phon}\}$ exhibits a small but measurable performance decrease from $\text{CNN}_{\text{text}}\{\text{str}\}$, possibly due to phonetic features capturing similar regularities, while also depending on the quality of preprocessing tools and the rule-based phonetic algorithm being relevant for our song-based dataset. The $\text{CNN}_{\text{text}}\{\text{str, phon, lsyn}\}$ model that combines the different textual SSMs yields a performance comparable to $\text{CNN}_{\text{text}}\{\text{str}\}$.

In addition, we test the performance of several line-based features on our dataset. Most notably, the n-grams feature provides a significant performance improvement producing the best model. Note that adding the line length feature to any CNN_{text} model does not increase performance.

To show the portability of our method to bigger and more heterogeneous datasets, we ran the CNN model on the WASABI dataset, obtaining results that are very close to the ones obtained for the MLDB dataset: precision: 67.4% for precision, 67.3% recall, and 67.4% f-score using the $\text{CNN}_{\text{text}}\{\text{str}\}$ model.

Results differ significantly based on genre. We split the MLDB dataset with genre annotations into training and test, trained on all genres, and tested on each genre separately. In Table 2.13 we report the performances of the $\text{CNN}_{\text{text}}\{\text{str}\}$ on lyrics of different genres. Songs belonging to genres such as Country, Rock or Pop, contain recurrent structures with repeating patterns, which are more easily detectable by the CNN_{text} algorithm. Therefore, they show significantly better performance. On the other hand, the performance on genres such as Hip Hop or Rap, is much worse.

The **results of our experiments with multimodal lyrics lines on the DALI dataset** are depicted in Table 2.14. The random baseline and the different line length baselines reach a

Model	*Features*	P	R	F_1
Random baseline	n/a	18.6	18.6	18.6
Line length baseline	text line length	16.7	52.8	25.4
Handcrafted filters	RPF (our replication)	48.2	67.8	56.3
	RPF [473]	56.1	59.4	57.7
	RPF + n-grams	57.4	61.2	59.2
CNN_{text}	{str}	70.4	63.0	66.5
	{phon}	75.9	55.6	64.2
	{lsyn}	74.8	50.0	59.9
	{str, phon, lsyn}	74.1	60.5	66.6
	{str, phon, lsyn} + n-grams	72.1	63.3	**67.4**

Table 2.12: Results with unimodal lyrics lines on MLDB dataset in terms of Precision (P), Recall (R) and F_1 in %.

Genre	*Lyrics[#]*	P	R	F_1
Rock	6011	73.8	57.7	64.8
Hip Hop	5493	71.7	43.6	<u>54.2</u>
Pop	4764	73.1	61.5	66.6
RnB	4565	71.8	60.3	65.6
Alternative Rock	4325	76.8	60.9	67.9
Country	3780	74.5	66.4	**70.2**
Hard Rock	2286	76.2	61.4	67.7
Pop Rock	2150	73.3	59.6	65.8
Indie Rock	1568	80.6	55.5	65.6
Heavy Metal	1364	79.1	52.1	63.0
Southern Hip Hop	940	73.6	34.8	<u>47.0</u>
Punk Rock	939	80.7	63.2	**70.9**
Alternative Metal	872	77.3	61.3	68.5
Pop Punk	739	77.3	68.7	**72.7**
Soul	603	70.9	57.0	63.0
Gangsta Rap	435	73.6	35.2	<u>47.7</u>

Table 2.13: Results with unimodal lyrics lines. $\text{CNN}_{\text{text}}\{str\}$ model performances across musical genres in the MLDB dataset in terms of Precision (P), Recall (R) and F_1 in %. Underlined are the performances on genres with less repetitive text. Genres with highly repetitive structure are in bold.

performance of 15.5%-33.5% F_1. Interestingly, the audio-based line length (33.5% F_1) is more indicative of the lyrics segmentation than the text-based line length (25.0% F_1).[24]

The model $\text{CNN}_{\text{text}}\{\text{str}\}$ performs with 70.8% F_1 similarly to the $\text{CNN}_{\text{text}}\{\text{str}\}$ model from the first experiment (66.5% F_1). The models use the exact same SSM_{str} feature and hyperparameters, but another lyrics corpus (DALI instead of MLDB). We believe that as DALI was assembled from karaoke singing instances, it likely contains more repetitive song texts that are easier to segment using the employed method. Note that the DALI dataset is too small to allow a genre-wise comparison as we did in the previous experiment using the MLDB dataset.

The $\text{CNN}_{\text{audio}}$ models perform similarly well than the CNN_{text} models. $\text{CNN}_{\text{audio}}\{\text{mfcc}\}$ reaches 65.3% F_1, while $\text{CNN}_{\text{audio}}\{\text{chr}\}$ results in 63.9% F_1. The model $\text{CNN}_{\text{audio}}\{\text{mfcc, chr}\}$ performs with 70.4% F_1 significantly ($p < .001$) better than the models that use only one of the features. As the mfcc feature models timbre and instrumentation, whilst the chroma feature models melody and harmony, they provide complementary information to the $\text{CNN}_{\text{audio}}$ model which increases its performance.

Most importantly, the CNN_{mult} models combining text- with audio-based features constantly outperform the CNN_{text} and $\text{CNN}_{\text{audio}}$ models. $\text{CNN}_{\text{mult}}\{\text{str, mfcc}\}$ and $\text{CNN}_{\text{mult}}\{\text{str, chr}\}$ achieve a performance of 73.8% F_1 and 74.5% F_1, respectively - this is significantly ($p < .001$) higher compared to the 70.8% (70.4%) F_1 of the best CNN_{text} ($\text{CNN}_{\text{audio}}$) model. Finally, the overall best performing model is a combination of the best CNN_{text} and $\text{CNN}_{\text{audio}}$ models and delivers 75.3% F_1. $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$ is the only model to significantly ($p < .05$) outperform all other models in all three evaluation metrics: precision, recall, and F_1. Note, that all CNN_{mult} models outperform all CNN_{text} and $\text{CNN}_{\text{audio}}$ models significantly ($p < .001$) in recall.

We perform an ablation test on the alignment quality. For this, we train CNN-based models with those feature sets that performed best on the Q^+ part of DALI. For each modality (text, audio, mult), i.e. $\text{CNN}_{\text{text}}\{\text{str}\}$, $\text{CNN}_{\text{audio}}\{\text{mfcc, chr}\}$, and $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$, we train a model for each feature set on each partition of DALI (Q^+, Q^0, Q^-). We always test our models on the same alignment quality they were trained on. The alignment quality ablation results are depicted in Table 2.15. We find that independent of the modality (text, audio, mult.), all models perform significantly ($p < .001$) better with higher alignment quality. The effect of modality on segmentation performance (F_1) is as follows: on all datasets we find $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$ to significantly ($p < .001$) outperform both $\text{CNN}_{\text{text}}\{\text{str}\}$ and $\text{CNN}_{\text{audio}}\{\text{mfcc, chr}\}$. Further, $\text{CNN}_{\text{text}}\{\text{str}\}$ significantly ($p < .001$) outperforms $\text{CNN}_{\text{audio}}\{\text{mfcc, chr}\}$ on the Q^0 and Q^- dataset, whereas this does not hold on the Q^+ dataset ($p \geq .05$).

2.4.5 Error analysis

An SSM for a Rap song is depicted in Figure 2.15. As texts in this genre are less repetitive, the SSM-based features are less reliable to determine a song's structure. Moreover, when returning to the introductory example in Figure 2.12, we observe that verses (the V_i) and bridges (the B_i) are not detectable when looking at the text representation only (see Figure 2.12, middle). The reason is that these verses have different lyrics. However, as these parts share the same melody, highlighted rectangles are visible in the melody structure.

Indeed, we found our bimodal segmentation model to produce significantly ($p < .001$) better segmentations (75.3% F_1) compared to the purely text-based (70.8% F_1) and audio-based models (70.4% F_1). The increase in F_1 stems from both increased precision and recall. The model increase in precision is observed as CNN_{mult} often produces less false positive segment borders, i.e. the model delivers less noisy results. We observe an increase in recall in two ways: first, CNN_{mult} sometimes detects a combination of the borders detected by CNN_{text} and

[24]Note that adding line length features to any CNN-based model does not increase performance.

Model	*Features*	P	R	F_1
Random baseline	n/a	15.7	15.7	15.7
Line length baselines	text length	16.6	51.8	25.0
	audio length	22.7	63.8	33.5
	text length + audio length	22.6	63.0	33.2
CNN_{text}	{str}	78.7	64.2	70.8
$\text{CNN}_{\text{audio}}$	{mfcc}	79.3	55.9	65.3
	{chr}	76.8	54.7	63.9
	{mfcc, chr}	79.2	63.8	70.4
CNN_{mult}	{str, mfcc}	80.6	**69.0**	73.8
	{str, chr}	82.5	**69.0**	74.5
	{str, mfcc, chr}	**82.7**	**70.3**	**75.3**

Table 2.14: Results with multimodal lyrics lines on the Q^+ dataset in terms of Precision (P), Recall (R) and F_1 in %. Note that the CNN_{text}{str} model is the same configuration as in Table 2, but trained on different dataset.

Dataset	*Model*	*Features*	P	R	F_1
Q^+	CNN_{text}	{str}	78.7	64.2	70.8
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	79.2	63.8	70.4
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	82.7	70.3	75.3
Q^0	CNN_{text}	{str}	73.6	54.5	62.8
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	74.9	48.9	59.5
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	75.8	59.4	66.5
Q^-	CNN_{text}	{str}	67.5	30.9	41.9
	$\text{CNN}_{\text{audio}}$	{mfcc, chr}	66.1	24.7	36.1
	$\text{CNN}_{\text{mult.}}$	{str, mfcc, chr}	68.0	35.8	46.7

Table 2.15: Results with multimodal lyrics lines for the alignment quality ablation test on the datasets Q^+, Q^0, Q^- in terms of Precision (P), Recall (R) and F_1 in %.

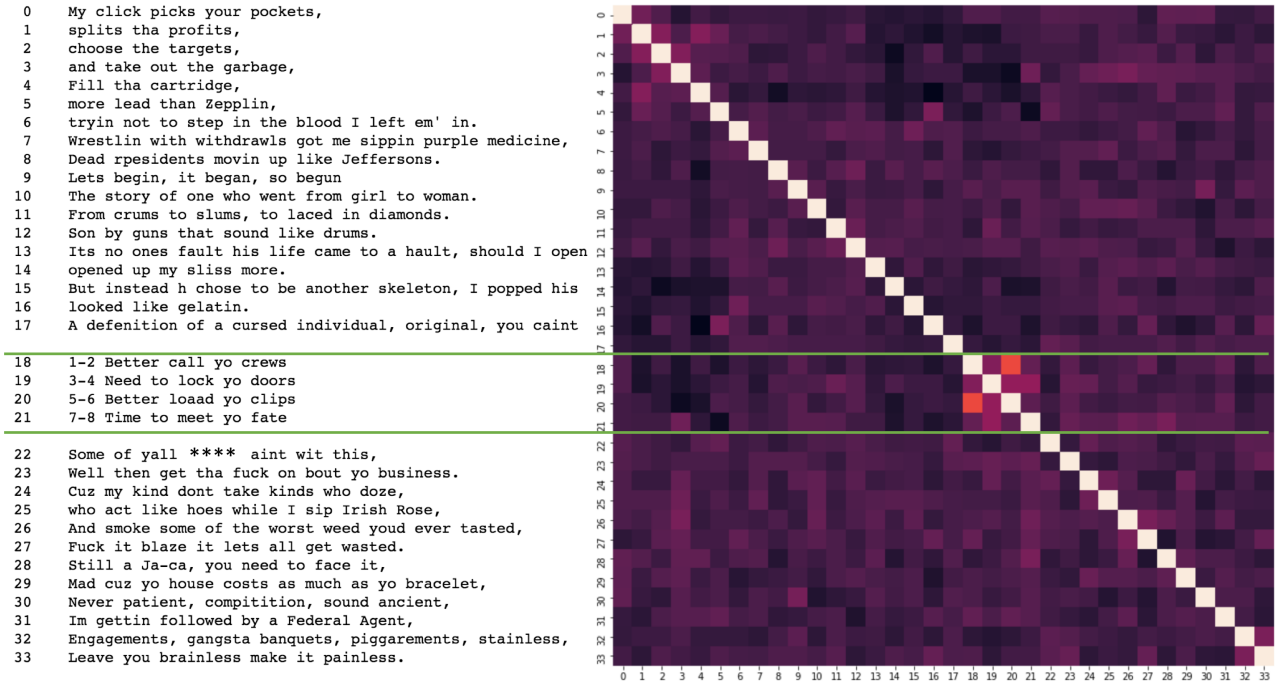


Figure 2.15: Example SSM computed from textual similarity sim_{str}. As common for Rap song texts, there is no chorus (diagonal stripe parallel to main diagonal). However, there is a highly repetitive musical state from line 18 to 21 indicated by the corresponding rectangle in the SSM spanning from (18,18) to (21,21). ("Meet Your Fate" by Southpark Mexican, MLDB-ID: 125521)

$\text{CNN}_{\text{audio}}$. Secondly, there are cases where CNN_{mult} detects borders that are not recalled in either of CNN_{text} or $\text{CNN}_{\text{audio}}$.

Segmentation algorithms that are based on exploiting patterns in an SSM, share a common limitation: non-repeated segments are hard to detect as they do not show up in the SSM. Note, that such segments are still occasionally detected indirectly when they are surrounded by repeated segments. Furthermore, a consecutively repeated pattern such as C_2-C_3-C_4 in Figure 2.12 is not easily segmentable as it could potentially also form one ($C_2C_3C_4$) or two (C_2-C_3C_4 or C_2C_3-C_4) segments. Another problem is that of inconsistent classification inside of a song: sometimes, patterns in the SSM that look the same to the human eye are classified differently. Note, however that on the pixel level there is a difference, as the inference in the used CNN is deterministic. This is a phenomenon similar to adversarial examples in image classification (same intension, but different extension).

We now analyze the predictions of our different models for the example song given in Figure 2.12. We compare the predictions of the following three different models: the text-based model $\text{CNN}_{\text{text}}\{\text{str}\}$ (visualized in Figure 2.12 as the left SSM called "repetitive lyrics structure"), the audio-based model $\text{CNN}_{\text{audio}}\{\text{chr}\}$ (visualized in Figure 2.12 as the right SSM called "repetitive melody structure"), and the bimodal model $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$. Starting with the first chorus, C_1, we find it to be segmented correctly by both $\text{CNN}_{\text{text}}\{\text{str}\}$ and $\text{CNN}_{\text{audio}}\{\text{chr}\}$. As previously discussed, consecutively repeated patterns are hard to segment and our text-based model indeed fails to correctly segment the repeated chorus (C_2-C_3-C_4). The audio-based model $\text{CNN}_{\text{audio}}\{\text{chr}\}$ overcomes this limitation and segments the repeated chorus correctly. Finally, we find that in this example both the text-based and the audio-based models fail to segment the verses (the V_i) and bridges (the B_i) correctly. The $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$ model manages to detect the bridges and verses in our example.

Note that adding more features to a model does not always increase its ability to detect segment borders. While in some examples, the $\text{CNN}_{\text{mult}}\{\text{str, mfcc, chr}\}$ model detects segment borders that were not detected in any of the models $\text{CNN}_{\text{text}}\{\text{str}\}$ or $\text{CNN}_{\text{audio}}\{\text{mfcc, chr}\}$, there are also examples where the bimodal model does not detect a border that is detected by both the text-based and the audio-based models.

2.5 Song lyrics summarization inspired by audio thumbnailing

Automatic text summarization is the task of producing a concise and fluent summary while preserving key information content and overall meaning of a text [7]. Numerous approaches have been developed to address this task and applied widely in various domains including news articles [107], scientific papers [309], web content as blogs [231], customer reviews [369] and social media messages [218].

Just as we may need to summarize a story, we may also need to summarize song lyrics, for instance to produce adequate snippets for a search engine dedicated to an online song collection or for music digital libraries. From a linguistic point of view however, lyrics are a very peculiar genre of document and generic summarization methods may not be appropriate when the input for summarization comes from a specific domain or type of genre as songs are [342]. Compared to news documents, for instance, lyrics have a very different structure. Given the repeating forms, peculiar structure (e.g., the segmentation into verse, chorus, etc.) and other unique characteristics of song lyrics, we need the summarization algorithms to take advantage of these additional elements to more accurately identify relevant information in song lyrics. But just as such characteristics enable the exploration of new approaches, other characteristics make the application of summarization algorithms very challenging, as the presence of repeated lines, the discourse structure that strongly depends on the interrelation of music and words in the melody composition, the heterogeneity of musical genres each featuring peculiar styles and wording [71], and simply the fact that not all songs tell a story.

In this direction, this work focuses on the following research questions: *What is the impact of the context in summarizing song lyrics?*. This question is broken down into two sub questions: 1) *How do generic text summarization methods perform over lyrics?* and 2) *Can such peculiar context be leveraged to identify relevant sentences to improve song text summarization?* To answer our research questions, we experiment with generic unsupervised state-of-the-art text summarization methods (i.e. TextRank, and a topic distribution based method) to perform lyrics summarization, and show that adding contextual information helps such models to produce better summaries. Specifically, we enhance text summarization approaches with a method inspired by audio thumbnailing techniques, that leverages the repetitive structure of song texts to improve summaries. We show how summaries that take into account the audio nature of the lyrics outperform the generic methods according to both an automatic evaluation over 50k lyrics, and judgments of 26 human subjects.

As introduced in the previous section, song texts are arranged in segments and lines. For instance the song text depicted in Figure 2.16 consists of 8 segments and 38 lines. Given a song text S consisting of n lines of text, $S = (x_1, ..., x_n)$, we define the task of *extractive lyrics summarization* as the task of producing a concise summary *sum* of the song text, consisting of a subset of the original text lines: $sum(S) \subseteq S$, where usually $|sum(S)| << |S|$. We define the goal of a summary as to preserve key information and the overall meaning of a song text. To address this task, we apply the following methods from the literature: the popular graph-based summarizer TextRank; an adaptation of a topic-based method (TopSum). Moreover, we

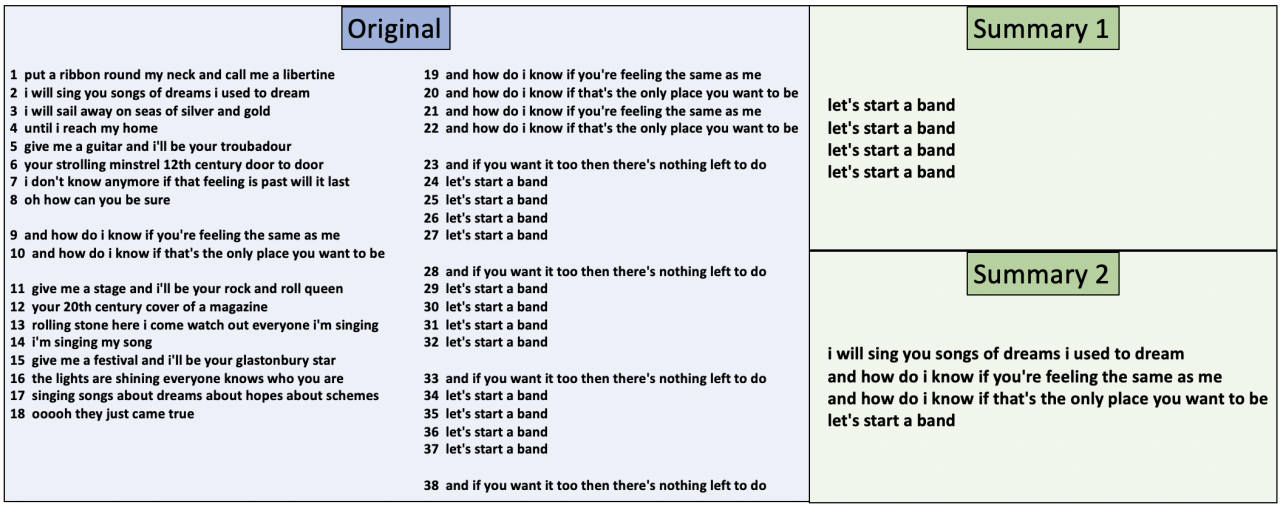


Figure 2.16: Song text of "Let's start a band" by Amy MacDonald along with two example summaries.

introduce a method inspired by audio thumbnailing (which we dub Lyrics Thumbnail) which aims at creating a summary from the most representative parts of the original song text. While for TextRank we rely on the off-the-shelf implementation of [33], in the following we describe the other two methods.

2.5.1 TopSum

We implement a simple topic-based summarization model that aims to construct a summary whose topic distribution is as similar as possible to that of the original text. Following [249], we train a topic model by factorizing a tf-idf-weighted term-document matrix of a song text corpus using non-negative matrix factorization into a term-topic and a topic-document matrix. Given the learnt term-topic matrix, we compute a topic vector t for each new document (song text). In order to treat t as a (pseudo-) probability distribution over latent topics t_i, we normalize t by applying $\lambda t.t/\sum_{t_i \in t} t_i$ to it. Given the distributions over latent topics for each song text, we then incrementally construct a summary by greedily adding one line from the original text at a time (same mechanism as in KLSum algorithm in [210]); that line x^* of the original text that minimizes the distance between the topic distribution t_S of the original text S and the topic distribution of the incremental summary $sum(S)$:

$$x^* = \underset{x \in (S \setminus sum(S))}{\operatorname{argmin}} \{W(t_S,\ t_{sum(S)+x})\}$$

W is the Wasserstein distance [457] and is used to measure the distance between two probability distributions (an alternative to Jensen-Shannon divergence [291]).

2.5.2 Lyrics Thumbnail

Inspired by [243], we transfer their fitness measure for audio segments to compute the fitness of lyrics segments. Analog to an audio thumbnail, we define a Lyrics Thumbnail as the most representative and repetitive part of the song text. Consequently, it usually consists of (a part of) the chorus. In our corpus the segments are annotated (as double line breaks in the lyrics), so unlike in audio thumbnailing, we do not have to induce segments, but rather measure their fitness. In the following, we describe the fitness measure for lyrics segments and how we use this to produce a summary of the lyrics.

Lyrics Fitness Given a segmented song text $S = (S_1, ..., S_m)$ consisting of text segments S_i, where each S_i consists of $|S_i|$ text lines, we cluster the S_i into partitions of similar segments. For instance, the lyrics in Figure 2.16 consists of 8 segments and 38 lines and the cluster of chorus consists of $\{S_5, S_6, S_7\}$. The fitness Fit of the segment cluster $C \subseteq S$ is defined through the precision pr of the cluster and the coverage co of the cluster. pr describes how similar the segments in C are to each other while co is the relative amount of lyrics lines covered by C:

$$pr(C) = (\sum_{\substack{S_i,S_j \in C \\ i<j}} 1)^{-1} \cdot \sum_{\substack{S_i,S_j \in C \\ i<j}} sim(S_i, S_j)$$

$$co(C) = (\sum_{S_i \in S} |S_i|)^{-1} \cdot \sum_{S_i \in C} |S_i|$$

where sim is a normalized similarity measure between text segments. Fit is the harmonic mean between pr and co. The fitness of a segment S_i is defined as the fitness of the cluster to which S_i belongs:

$$\forall S_i \in C : Fit(S_i) = Fit(C) = 2\frac{pr(C) \cdot co(C)}{pr(C) + co(C)}$$

For lyrics segments without repetition the fitness is defined as zero. Based on the fitness Fit for segments, we define a fitness measure for a text line x. This allows us to compute the fitness of arbitrary summaries (with no or unknown segmentation). If the text line x occurs $f_i(x)$ times in text segment S_i, then its line fitness fit is defined as:

$$fit(x) = (\sum_{S_i \in S} f_i(x))^{-1} \cdot \sum_{S_i \in S} f_i(x) \cdot Fit(S_i)$$

Fitness-Based Summary Analog to [243]'s audio thumbnails, we create fitness-based summaries for a song text. A *Lyrics Double Thumbnail* consists of two segments: one from the fittest segment cluster (usually the chorus), and one from the second fittest segment cluster (usually the bridge).[25] If the second fittest cluster has a fitness of 0, we generate a *Lyrics Single Thumbnail* solely from the fittest cluster (usually the chorus). If the thumbnail generated has a length of k lines and we want to produce a summary of $p < k$ lines, we select the p lines in the middle of the thumbnail following [103]'s "Section-transition Strategy" that they find to capture the "hook" of the music more likely.[26]

2.5.3 Experimental setting

Dataset. From the WASABI corpus [317] we select a subset of 190k unique song texts with available genre information. As the corpus has spurious genres (416 different ones), we focus on the 10 most frequent ones in order to evaluate our methods dependent on the genre. We add 2 additional genres from the underrepresented Rap field (Southern Hip Hop and Gangsta Rap). The dataset contains 95k song lyrics.

To define the length of $sum(S)$, we rely on [35] that recommend to create audio thumbnails of the median length of the chorus on the whole corpus. We therefore estimate the median chorus length on our corpus by computing a Lyrics Single Thumbnail on each text, and we find the median chorus length to be 4 lines. Hence, we decide to generate summaries of such length for all lyrics and all summarization models to exclude the length bias in the methods

[25] We pick the first occurring representative of the segment cluster. Which segment to pick from the cluster is a potential question for future work.

[26] They also experiment with other methods to create a thumbnail, such as section initial or section ending.

comparison[27]. As the length of the lyrics thumbnail is lower-bounded by the length of the chorus in the song text, we keep only those lyrics with an estimated chorus length of at least 4. The final corpus of 12 genres consists of 50k lyrics with the following genre distribution: Rock: 8.4k, Country: 8.3k, Alternative Rock: 6.6k, Pop: 6.9k, R&B: 5.2k, Indie Rock: 4.4k, Hip Hop: 4.2k, Hard Rock: 2.4k, Punk Rock: 2k, Folk: 1.7k, Southern Hip Hop: 281, Gangsta Rap: 185.

Models and Configurations. We create summaries using the three summarization methods previously described, i.e. a graph-based (TextRank), a topic-based (TopSum), and fitness-based (Lyrics Thumbnail) method, plus two additional combined models (described below). While the Lyrics Thumbnail is generated from the full segment structure of the lyrics including its duplicate lines, all other models are fed with unique text lines as input (i.e. redundant lines are deleted). This is done to produce less redundant summaries, given that for instance, TextRank scores each duplicate line the same, hence it may create summaries with all identical lines. TopSum can suffer from a similar shortcoming: if there is a duplicate line close to the ideal topic distribution, adding that line again will let the incremental summary under construction stay close to the ideal topic distribution. All models were instructed to produce summaries of 4 lines, as this is the estimated median chorus length in our corpus. The summary lines were arranged in the same order they appear in the original text.[28] We use the TextRank implementation[29] of [33] without removing stop words (lyrics lines in input can be quite short, therefore we avoid losing all content of the line if removing stop words). The topic model for TopSum is built using non-negative matrix factorization with scikit-learn[30] [370] for 30 topics on the full corpus of 190k lyrics.[31] For the topical distance, we only consider the distance between the 3 most relevant topics in the original text, following the intuition that one song text usually covers only a small amount of topics. The Lyrics Thumbnail is computed using String-based distance between text segments to facilitate clustering. This similarity has been shown in [473] to indicate segment borders successfully. In our implementation, segments are clustered using the DBSCAN [164] algorithm.[32] We also produce two summaries by combining TextRank + TopSum and TextRank + TopSum + Lyrics Thumbnail, to test if summaries can benefit from the complementary perspectives the three different summarization methods take.

Model Combination. For any lyrics line, we can obtain a score from each of the applied methods. TextRank provides a score for each line, TopSum provides a distance between the topic distributions of an incremental summary and the original text, and fit provides the fitness of each line. We treat our summarization methods as blackboxes and use a simple method to combine the scores the different methods provide for each line. Given the original text separated into lines $S = (x_1, ..., x_n)$, a summary is constructed by greedily adding one line x^* at a time to the incremental summary $sum(S) \subseteq S$ such that the sum of normalized ranks of all scores is minimal:

$$x^* = \text{argmin} \bigcup_x \{\sum_A \text{R}_\text{A}(x)\}$$

Here $x \in (S \setminus sum(S))$ and $A \in \{\text{TextRank}, \text{TopSum}, \text{fit}\}$. The normalized rank $\text{R}_\text{A}(x)$ of the score that method A assigns to line x is computed as follows: first, the highest scores[33] are assigned rank 0, the second highest scores get rank 1, and so forth. Then the ranks are linearly scaled to the [0,1] interval, so each sum of ranks $\sum_A \text{R}_\text{A}(x)$ is in [0,3].

[27] We leave the study of other measures to estimate the summary length to future work.
[28] In case of repeated parts, the first position of each line was used as original position.
[29] https://github.com/summanlp/textrank
[30] https://scikit-learn.org
[31] loss='kullback-leibler'
[32] eps=0.3, min_samples=2
[33] In the case of topical distance, a "higher score" means a lower value.

Model Nomenclature For abbreviation, we call the TextRank model henceforth M_r, the TopSum model M_s, the fitness-based summarizer M_f, model combinations M_{rs} and M_{rsf}, respectively.

2.5.4 Evaluation

We evaluate the quality of the produced lyrics summary both soliciting human judgments on the goodness and utility of a given summary, and through an automatic evaluation of the summarization methods to provide a comprehensive evaluation.

Human Evaluation. We performed human evaluation of the different summarization methods introduced before by asking participants to rate the different summaries presented to them by specifying their agreement / disagreement according to the following standard criteria [366]:
Informativeness: The summary contains the main points of the original song text.
Non-redundancy: The summary does not contain duplicate or redundant information.
Coherence: The summary is fluent to read and grammatically correct.
Plus one additional criterion coming from our definition of the lyrics summarization task:
Meaning: The summary preserves the meaning of the original song text.

An experimental psychologist expert in Human Computer Interaction advised us in defining the questionnaire and setting up the experiment. 26 participants - 12 nationalities, 18 men, 8 women, aged from 21 to 59 - were taking a questionnaire (Google Forms), consisting of rating 30 items with respect to the criteria defined before on a Likert scale from 1 (low) to 5 (high). Each participant was presented with 5 different summaries - each produced by one of the previously described summarization models - for 6 different song texts. Participants were given example ratings for the different criteria in order to familiarize them with the procedure. Then, for each song text, the original song text along with its 5 summaries were presented in random order and had to be rated according to the above criteria. For the criterion of Meaning, we asked participants to give a short explanation in free text for their score. The selected 6 song texts[34] have a minimum and a median chorus length of 4 lines and are from different genres, i.e. Pop/Rock (4), Folk (1) and Rap (1), similar to our corpus genre distribution. Song texts were selected from different lengths (18-63 lines), genders of singer (3 male, 3 female), topics (family, life, drugs, relationship, depression), and mood (depressive, angry, hopeful, optimistic, energetic). The artist name and song title were not shown to the participants.

Figure 2.17 shows the ratings obtained for each criterion. We examine the significant differences between the models performances by performing a paired two-tailed t-test. The significance levels are: $0.05^{*}, 0.01^{**}, 0.001^{***}$, and $n.s.$ First, Informativeness and Meaning are rated higher** for the combined model M_{rs} compared to the single models M_r and M_s. Combining all three models improves the summaries further: both for Informativeness and Meaning the model M_{rsf} is rated higher*** than M_{rs}. Further, summaries created by M_{rsf} are rated higher*** in Coherence than summaries from any other model - except from M_f ($n.s.$ difference). Summaries are rated on the same level ($n.s.$ differences) for Non-redundancy in all but the M_r and M_f summaries, which are perceived as lower*** in Non-redundancy than all others. Note, how the model M_{rsf} is more stable than all others by exhibiting lower standard deviations in all criteria except Non-redundancy. The criteria Informativeness and Meaning are highly correlated (Pearson correlation coefficient 0.84). Correlations between other criteria range between 0.29 and 0.51.

[34] "Pills N Potions" by Nicki Minaj, "Hurt" by Nine Inch Nails, "Real to me" by Brian McFadden, "Somebody That I Used To Know" by Gotye, "Receive" by Alanis Morissette, "Let's Start A Band" by Amy MacDonald

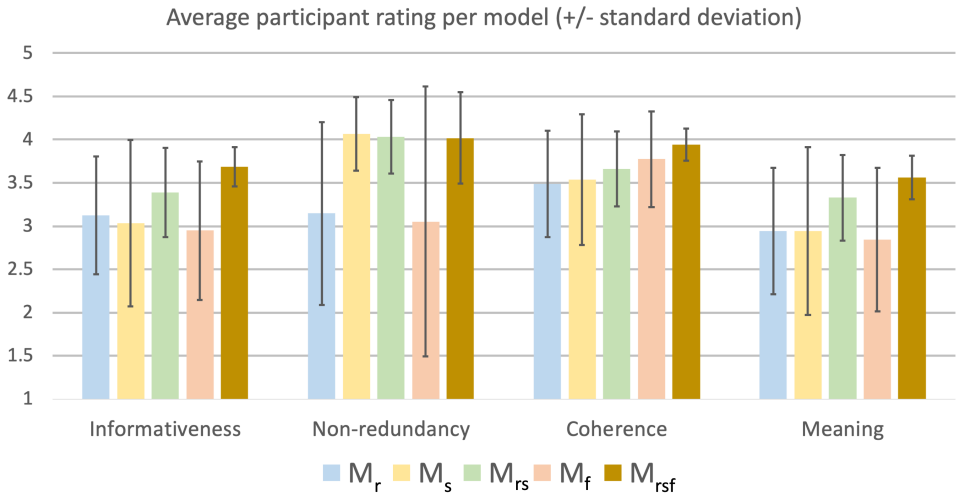


Figure 2.17: Human ratings per summarization model in terms of average and standard deviation.

Evaluation criterion	**Genre**	M_r	M_s	M_{rs}	M_f	M_{rsf}	original text
Distributional Semantics [%]	Rock / Pop	92	100	97	90	93	n/a
	Rap	94	100	99	86	92	
	$\sum$	92	100	98	90	93	
Topical [%]	Rock / Pop	44	100	76	41	64	n/a
	Rap	58	100	80	48	66	
	$\sum$	46	**100**	77	<u>42</u>	**64**	
Coherence [%]	Rock / Pop	110	95	99	99	100	100
	Rap	112	115	112	107	107	
	$\sum$	110	97	101	100	101	
Lyrics fitness [%]	Rock / Pop	71	53	63	201	183	100
	Rap	0	0	0	309	249	
	$\sum$	62	<u>47</u>	55	**214**	**191**	

Table 2.16: Automatic evaluation results for the 5 summarization models and 2 genre clusters. Distributional Semantics and Topical are relative to the best model (=100%), Coherence and Fitness to the original text (=100%).

Overall, leveraging the Lyrics Fitness in a song text summary improves summary quality. Especially with respect to the criteria that, we believe, indicate the summary quality the most - Informativeness and Meaning - the M_{rsf} method is significantly better performing and more consistent.

Figure 2.16 shows an example song text and example summaries from the experiment. Summary 1 is generated by M_f and consists of the chorus. Summary 2 is made by the method M_{rsf} and has relevant parts of the verses and the chorus, and was rated much higher in Informativeness and Meaning. We analyzed the free text written by the participants to comment on the Meaning criterion, but no relevant additional information was provided (the participants mainly summarized their ratings).

Automatic Evaluation. We computed four different indicators of summary quality on the dataset of 50k songs described before. Three of the criteria use the similarity between probability distributions P, Q, which means we compute the Wasserstein distance between P and Q

(cf. Subsection 2.5.1) and apply $\lambda x.\ x^{-1}$ to it.[35] The criteria are:

Distributional Semantics: similarity between the word distributions of original and summary, cf. [291]. We give results relative to the similarity of the best performing model (=100%).

Topical: similarity between the topic distributions of original and summary. Restricted to the 3 most relevant topics of the original song text. We give results relative to the similarity of the best performing model (=100%).

Coherence: average similarity between word distributions in consecutive sentences of the summary, cf. [409]. We give results relative to the coherence of the original song text (=100%).

Lyrics fitness: average line-based fitness fit of the lines in the summary. We give results relative to the Lyrics fitness of the original song text (=100%).

When evaluating each of the 12 genres, we found two clusters of genres to behave very similarly. Therefore, we report the results for these two groups: the *Rap* genre cluster contains Hip Hop, Southern Hip Hop, and Gangsta Rap. The *Rock / Pop* cluster contains the 9 other genres. Results of the different automatic evaluation metrics are shown in Table 2.16. Distributional Semantics metrics have previously been shown [291, 409] to highly correlate with user responsiveness judgments. We would expect correlations of this metric with Informativeness or Meaning criteria therefore, as those criteria are closest to responsiveness, but we have found no large differences between the different models for this criterion. The summaries of the M_s model have the highest similarity to the original text and the M_f have the lowest similarity of 90%. The difference between the highest and lowest values are low.

For the Topical similarity, the results are mostly in the same order as the Distributional Semantics ones, but with much larger differences. While the M_s model reaches the highest similarity, this is a self-fulfilling prophecy, as summaries of M_s were generated with the objective of maximizing topical similarity. The other two models that incorporate M_s (M_{rs} and M_{rsf}), show a much higher topical similarity to the original text than M_r and M_f.

Coherence is rated best in M_r with 110%. All other models show a coherence close to that of the original text - between 97% and 101%. We believe that the increased coherence of M_r is not linguistically founded, but merely algorithmic. M_r produces summaries of the most central sentences in a text. The centrality is using the concept of sentence similarity. Therefore, M_r implicitly optimizes for the automatic evaluation metric of coherence, based on similar consecutive sentences. Sentence similarity seems to be insufficient to predict human judgments of coherence in this case.

As might be expected, methods explicitly incorporating the Lyrics fitness produce summaries with a fitness much higher than the original text - 214% for the M_f and 191% for the M_{rsf} model. The methods not incorporating fitness produce summaries with much lower fitness than the original - M_r 62%, M_s 47%, and M_{rs} 55%. In the Rap genre this fitness is even zero, i.e. summaries (in median) contain no part of the chorus.

Overall, no single automatic evaluation criterion was able to explain the judgments of our human participants. However, considering Topical similarity and fitness together gives us a hint. The model M_f has high fitness (214%), but low Topical similarity (42%). The M_s model has the highest Topical similarity (100%), but low fitness (47%). M_{rsf} might be preferred by humans as it strikes a balance between Topical similarity (64%) and fitness (191%). Hence, M_{rsf} succeeds in capturing lines from the most relevant parts of the lyrics, such as the chorus, while jointly representing the important topics of the song text.

[35] This works as we always deal with distances > 0.

2.6 Enriching the WASABI Song Corpus with lyrics annotations

Let's imagine the following scenario: following David Bowie's death, a journalist plans to prepare a radio show about the artist's musical career to acknowledge his qualities. To discuss the topic from different angles, she needs to have at her disposal the artist biographical information to know the history of his career, the song lyrics to know what he was singing about, his musical style, the emotions his songs were conveying, live recordings and interviews. Similarly, streaming professionals such as Deezer, Spotify, Pandora or Apple Music aim at enriching music listening with artists' information, to offer suggestions for listening to other songs/albums from the same or similar artists, or automatically determining the emotion felt when listening to a track to propose coherent playlists to the user. To support such scenarios, the need for rich and accurate musical knowledge bases and tools to explore and exploit this knowledge becomes evident.

For this reason, we integrate the results of all the methods for lyrics processing we developed into the WASABI Song Corpus, a large corpus of songs (2.10M songs, 1.73M with lyrics) enriched with metadata extracted from music databases on the Web, and resulting from the processing of song lyrics and from audio analysis. The corpus contains songs in 36 different languages, even if the vast majority are in English. As for the songs genres, the most common ones are Rock, Pop, Country and Hip Hop.

More specifically, while an overview of the goals of the WASABI project supporting the dataset creation and the description of a preliminary version of the dataset can be found in [317], in this section we focus on the description of the methods we proposed to annotate relevant information in the song lyrics. Given that lyrics encode an important part of the semantics of a song, we propose to label the WASABI dataset lyrics with their structure segmentation, the explicitness of the lyrics content, the salient passages of a song, the addressed topics and the emotions conveyed.

An analysis of the correlations among the above mentioned annotation layers reveals interesting insights about the song corpus. For instance, we demonstrate the change in corpus annotations diachronically: we show that certain topics become more important over time and others are diminished. We also analyze such changes in explicit lyrics content and expressed emotion.

2.6.1 The WASABI Song Corpus

In the context of the WASABI research project[36] that started in 2017, a two million song database has been built, with metadata on 77k artists, 208k albums, and 2.10M songs [317]. The metadata has been *i)* aggregated, merged and curated from different data sources on the Web, and *ii)* enriched by pre-computed or on-demand analyses of the lyrics and audio data.

We have performed various levels of analysis, and interactive Web Audio applications have been built on top of the output. For example, the TimeSide analysis and annotation framework have been linked [180] to make on-demand audio analysis possible. In connection with the FAST project[37], an offline chord analysis of 442k songs has been performed, and both an online enhanced audio player [367] and chord search engine [368] have been built around it. A rich set of Web Audio applications and plugins has been proposed [76, 77, 78], that allow, for example, songs to be played along with sounds similar to those used by artists. All these metadata, computational analyses and Web Audio applications have now been gathered in one

[36] http://wasabihome.i3s.unice.fr/
[37] http://www.semanticaudio.ac.uk

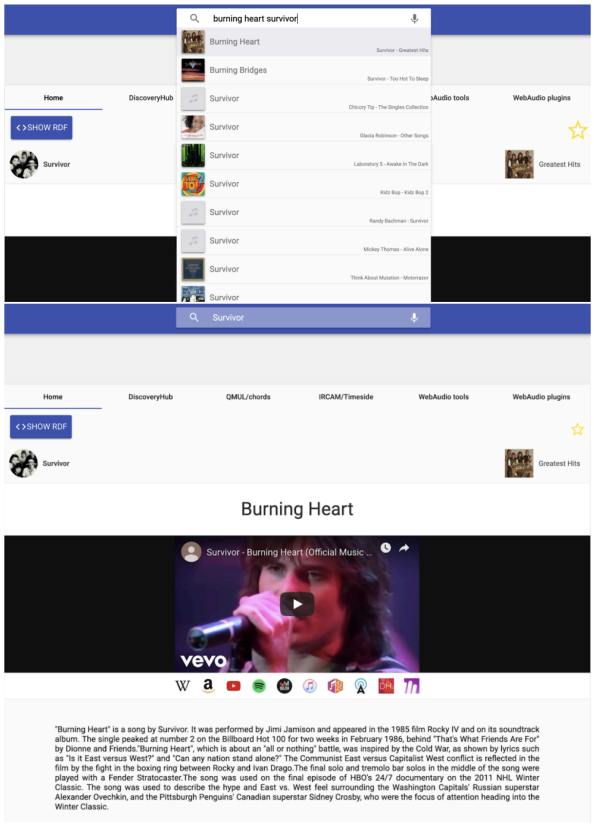


Figure 2.18: The WASABI Interactive Navigator.

easy-to-use web interface, the WASABI Interactive Navigator[38], illustrated[39] in Figure 2.18.

We have started building the WASABI Song Corpus by collecting for each artist the complete discography, band members with their instruments, time line, equipment they use, and so on. For each song we collected its lyrics from LyricWiki[40], the synchronized lyrics when available[41], the DBpedia abstracts and the categories the song belongs to, e.g., genre, label, writer, release date, awards, producers, artist and band members, the stereo audio track from Deezer, the unmixed audio tracks of the song, its ISRC, bpm and duration.

We matched the song ids from the WASABI Song Corpus with the ids from MusicBrainz, iTunes, Discogs, Spotify, Amazon, AllMusic, GoHear, YouTube. Figure 2.19 illustrates[42] all the data sources we have used to create the WASABI Song Corpus. We have also aligned the WASABI Song Corpus with the publicly available LastFM dataset[43], resulting in 327k tracks in our corpus having a LastFM id.

As of today, the corpus contains 1.73M songs with lyrics (1.41M unique lyrics). 73k songs have at least an abstract on DBpedia, and 11k have been identified as "classic songs" (they have been number one, or got a Grammy award, or have lots of cover versions). About 2k songs have a multi-track audio version, and on-demand source separation using open-unmix [430] or Spleeter [220] is provided as a TimeSide plugin.

Several Natural Language Processing methods have been applied to the lyrics of the songs included in the WASABI Song Corpus, as well as various analyses of the extracted information have been carried out. After providing some statistics on the WASABI corpus, the rest of the

[38] `http://wasabi.i3s.unice.fr/`
[39] Illustration taken from [79].
[40] `http://lyrics.wikia.com/`
[41] from `http://usdb.animux.de/`
[42] Illustration taken from [80].
[43] `http://millionsongdataset.com/lastfm/`

Figure 2.19: The datasources connected to the WASABI.

section describes the different annotations we added to the lyrics of the songs in the dataset. Based on the research we have conducted, the following lyrics annotations are added: lyrical structure (Section 2.6.2), summarization (Section 2.6.3), explicit lyrics (Section 2.6.4), emotion in lyrics (Section 2.6.5) and topics in lyrics (Section 2.6.6).

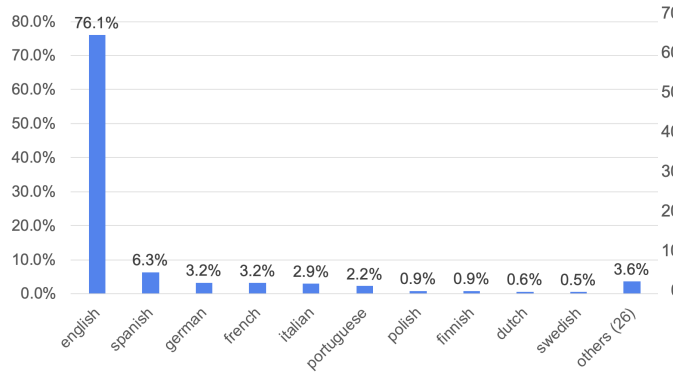
Statistics on the WASABI Song Corpus. This section summarizes key statistics on the corpus, such as the language and genre distributions, the songs coverage in terms of publication years, and then gives the technical details on its accessibility.

Language Distribution Figure 2.20a shows the distribution of the ten most frequent languages in our corpus.[44] In total, the corpus contains songs of 36 different languages. The vast majority (76.1%) is English, followed by Spanish (6.3%) and by four languages in the 2-3% range (German, French, Italian, Portugese). On the bottom end, Swahili and Latin amount to 0.1% (around 2k songs) each.

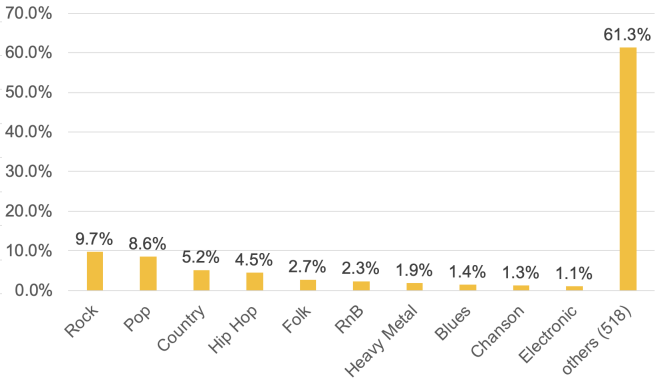
Genre distribution In Figure 2.20b we depict the distribution of the ten most frequent genres in the corpus.[45] In total, 1.06M of the titles are tagged with a genre. It should be noted that the genres are very sparse with a total of 528 different ones. This high number is partially due to many subgenres such as Alternative Rock, Indie Rock, Pop Rock, etc. which we omitted in Figure 2.20b for clarity. The most common genres are Rock (9.7%), Pop (8.6%), Country (5.2%), Hip Hop (4.5%) and Folk (2.7%).

Publication year Figure 2.20c shows the number of songs published in our corpus, by decade.[46] We find that over 50% of all songs in the WASABI Song Corpus are from the 2000s or later and only around 10% are from the seventies or earlier.

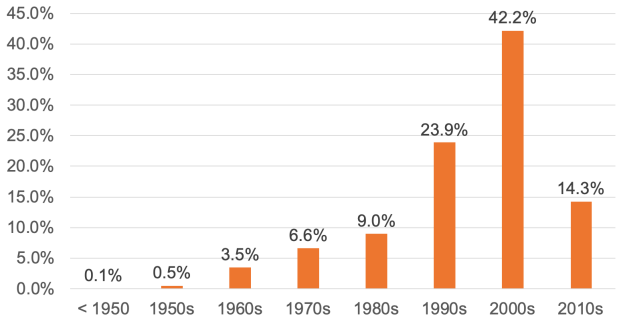
[44]Based on language detection performed on the lyrics.
[45]We take the genre of the album as ground truth since song-wise genres are much rarer.
[46]We take the album publication date as proxy since song-wise labels are too sparse.



(a) Language distribution (100% = 1.73M)



(b) Genre distribution (100% = 1.06M)



(c) Decade of publication distribution (100% = 1.70M)

Figure 2.20: Statistics on the WASABI Song Corpus

Accessibility of the WASABI Song Corpus The WASABI Interactive Navigator relies on multiple database engines: it runs on a MongoDB server altogether with an indexation by Elasticsearch and also on a Virtuoso triple store as a RDF graph database. It comes with a REST API[47] and an upcoming SPARQL endpoint. All the database metadata is publicly available[48] under a CC licence through the WASABI Interactive Navigator as well as programmatically through the WASABI REST API.

We provide the files of the current version of the WASABI Song Corpus, the models we have built on it as well as updates here: `https://github.com/micbuffa/WasabiDataset`.

Table 2.17 summarizes the most relevant annotations in our corpus.

2.6.2 Lyrics structure annotations

In Section 2.4 we presented a method to segment lyrics based on their repetitive structure in the form of a self-similarity matrix (SSM) [175].

In the WASABI Interactive Navigator, the line-based SSM of a song text can be visualized. It is toggled by clicking on the violet-blue square on top of the song text. For a subset of songs the color opacity indicates how repetitive and representative a segment is, based on the fitness metric that we proposed in Section 2.5 [172]. Note how in Figure 2.21, the segments 2, 4 and 7 are shaded more darkly than the surrounding ones. As highly fit (opaque) segments often coincide with a chorus, this is a first approximation of chorus detection. Given the variability in the set of structure types provided in the literature according to different genres [433, 71], rare attempts have been made in the literature to achieve a more complete semantic labelling,

[47] `https://wasabi.i3s.unice.fr/apidoc/`

[48] There is no public access to copyrighted data such as lyrics and full length audio files. Instructions on how to obtain lyrics are nevertheless provided and audio extracts of 30s length are available for nearly all songs.

Annotation	*Labels*	*Description*
Lyrics	1.73M	segments of lines of text
Languages	1.73M	36 different ones
Genre	1.06M	528 different ones
Last FM id	326k	UID
Structure	1.73M	SSM $\in \mathbb{R}^{n \times n}$ (n: length)
$\mathbb{S}$ocial tags	276k	$\mathbb{S} = \{$rock, joyful, 90s, ...$\}$
$\mathbb{E}$motion tags	87k	$\mathbb{E} \subset \mathbb{S} = \{$joyful, tragic, ...$\}$
Explicitness	715k	True (52k), False (663k)
Explicitness ♣	455k	True (85k), False (370k)
Summary♣	50k	four lines of song text
Emotion	16k	(valence, arousal) $\in \mathbb{R}^2$
Emotion♣	1.73M	(valence, arousal) $\in \mathbb{R}^2$
Topics♣	1.05M	Prob. distrib. $\in \mathbb{R}^{60}$
Total tracks	2.10M	diverse metadata

Table 2.17: Most relevant song-wise annotations in the WASABI Song Corpus. Annotations with ♣ are predictions of our models.

labelling the lyrics segments as Intro, Verse, Bridge, Chorus etc.

For each song text we provide an SSM based on a normalized character-based edit distance[49] on two levels of granularity to enable other researchers to work with these structural representations: line-wise similarity and segment-wise similarity.

2.6.3 Lyrics summary

Given the repeating forms, peculiar structure and other unique characteristics of song lyrics, in Section 2.5 we introduced a method for extractive summarization of lyrics that takes advantage of these additional elements to more accurately identify relevant information in song lyrics. Figure 2.22 shows another example summary of four lines length obtained with our proposed combined method. It is toggled in the WASABI Interactive Navigator by clicking on the green square on top of the song text.

The four-line summaries of 50k English used in our experiments are freely available within the WASABI Song Corpus; the Python code of the applied summarization methods is also available[50].

2.6.4 Explicit language in lyrics

On audio recordings, the Parental Advisory Label is placed in recognition of profanity and to warn parents of material potentially unsuitable for children. Nowadays, such labelling is carried out mainly manually on voluntary basis, with the drawbacks of being time consuming and therefore costly, error prone and partly a subjective task. In [171] we have tackled the task of automated explicit lyrics detection, based on the songs carrying such a label. We compared automated methods ranging from dictionary-based lookup to state-of-the-art deep neural networks to automatically detect explicit contents in English lyrics. More specifically, the dictionary-based methods rely on a swear word dictionary D_n which is automatically created from example explicit and clean lyrics. Then, we use D_n to predict the class of an unseen

[49] In our segmentation experiments we found this simple metric to outperform more complex metrics that take into account the phonetics or the syntax.

[50] `https://github.com/TuringTrain/lyrics\_thumbnailing`

Figure 2.21: Structure of the lyrics of "Everytime" by Britney Spears as displayed in the WASABI Interactive Navigator.

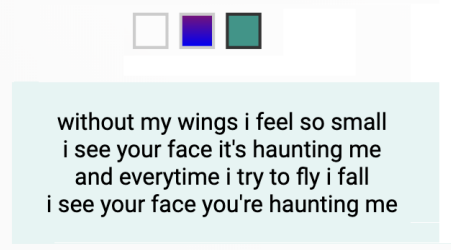


Figure 2.22: Summary of the lyrics of "Everytime" by Britney Spears as displayed in the WASABI Interactive Navigator.

Model	P	R	F_1
Majority Class	45.0	50.0	47.4
Dictionary Lookup	78.3	76.4	77.3
Dictionary Regression	76.2	81.5	78.5
Tf-idf BOW Regression	75.6	81.2	78.0
TDS Deconvolution	81.2	78.2	79.6
BERT Language Model	84.4	73.7	77.7

Table 2.18: Performance comparison of our different models. Precision (P), Recall (R) and f-score (F_1) in %.

song text in one of two ways: (i) the *Dictionary Lookup* simply checks if a song text contains words from D_n. (ii) the *Dictionary Regression* uses BOW made from D_n as the feature set of a logistic regression classifier. In the *Tf-idf BOW Regression* the BOW is expanded to the whole vocabulary of a training sample instead of only the explicit terms. Furthermore, the model *TDS Deconvolution* is a deconvolutional neural network [453] that estimates the importance of each word of the input for the classifier decision. In our experiments, we worked with 179k lyrics that carry gold labels provided by Deezer (17k tagged as explicit) and obtained the results shown in Table 2.18. We found the very simple *Dictionary Lookup* method to perform on par with much more complex models such as the *BERT Language Model* [143] as a text classifier. Our analysis revealed that some genres are highly overrepresented among the explicit lyrics. Inspecting the automatically induced explicit words dictionary reflects that genre bias. The dictionary of 32 terms used for the dictionary lookup method consists of around 50% of terms specific to the Rap genre, such as glock, gat, clip (gun-related), thug, beef, gangsta, pimp, blunt (crime and drugs). Finally, the terms holla, homie, and rapper are obviously no swear words, but highly correlated with explicit content lyrics.

Our corpus contains 52k tracks labelled as explicit and 663k clean (not explicit) tracks[51]. We have trained a classifier (77.3% f-score on test set) on the 438k English lyrics which are labelled and classified the remaining 455k previously untagged English tracks. We provide both the predicted labels in the WASABI Song Corpus and the trained classifier to apply it to unseen text.

2.6.5 Emotional description

In sentiment analysis the task is to predict if a text has a positive or a negative emotional valence. In the recent years, a transition from detecting sentiment (positive vs. negative valence) to more complex formulations of emotion detection (e.g., joy, fear, surprise) [332] has become more visible; even tackling the problem of emotion in context [105]. One family of emotion detection approaches is based on the valence-arousal model of emotion [399], locating every emotion in a two-dimensional plane based on its valence (positive vs. negative) and arousal (aroused vs. calm).[52] Figure 2.23 is an illustration of the valence-arousal model of Russell and shows exemplary where several emotions such as joyful, angry or calm are located in the plane. Manually labelling texts with multi-dimensional emotion descriptions is an inherently hard task. Therefore, researchers have resorted to distant supervision, obtaining gold labels from social tags from lastfm. These approaches [233, 97] define a list of social tags that are related to emotion, then project them into the valence-arousal space using an emotion lexicon [470, 331].

[51]Labels provided by Deezer. Furthermore, 625k songs have a different status such as unknown or censored version.

[52]Sometimes, a third dimension of dominance is part of the model.

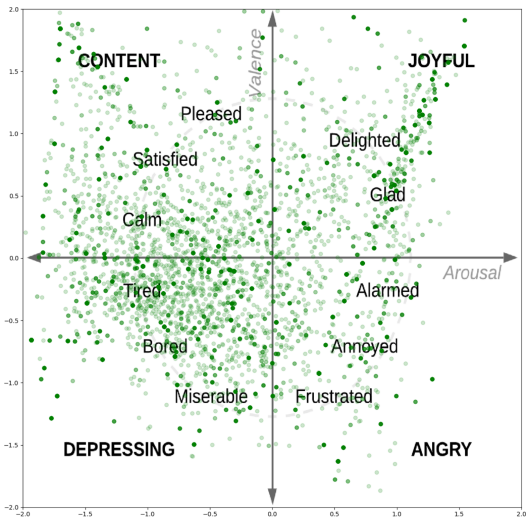


Figure 2.23: Emotion distribution in the corpus in the valence-arousal plane.

Recently, Deezer made valence-arousal annotations for 18,000 English tracks available[53] they have derived by the aforementioned method [139]. We aligned the valence-arousal annotations of Deezer to our songs. In Figure 2.23 the green dots visualize the emotion distribution of these songs.[54] Based on their annotations, we train an emotion regression model using BERT, with an evaluated 0.44/0.43 Pearson correlation/Spearman correlation for valence and 0.33/0.31 for arousal on the test set.

We integrated Deezer's labels into our corpus and also provide the valence-arousal predictions for the 1.73M tracks with lyrics. We also provide the last.fm social tags (276k) and emotion tags (87k entries) to facilitate researchers to build variants of emotion recognition models.

2.6.6 Topic Modelling

We built a topic model on the lyrics of our corpus using Latent Dirichlet Allocation (LDA) [56]. We determined the hyperparameters α, η and the topic count such that the coherence was maximized on a subset of 200k lyrics. We then trained a topic model of 60 topics on the unique English lyrics (1.05M).

We have manually labelled a number of more recognizable topics. Figures 2.25-2.29 illustrate these topics with word clouds[55] of the most characteristic words per topic. For instance, the topic Money contains words of both the field of earning money (job, work, boss, sweat) as well as spending it (pay, buy). The topic Family is both about the people of the family (mother, daughter, wife) and the land (sea, valley, tree). We provide the topic distribution of our LDA topic model for each song and make available the trained topic model to enable its application to unseen lyrics.

2.6.7 Diachronic corpus analysis

We examine the changes in the annotations over the course of time by grouping the corpus into decades of songs according to the distribution shown in Figure 2.20c.

[53] `https://github.com/deezer/deezer\_mood\_detection\_dataset`
[54] Depiction without scatterplot taken from [362]
[55] made with `https://www.wortwolken.com/`

Figure 2.24: Topic War

Figure 2.25: Topic Death

Figure 2.26: Topic Love

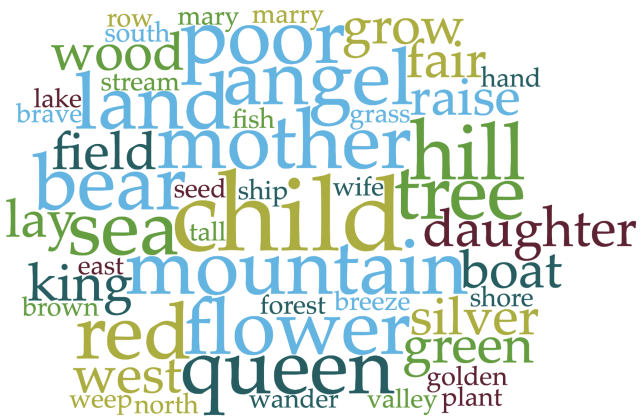


Figure 2.27: Topic Family

Figure 2.28: Topic Money

Figure 2.29: Topic Religion

Changes in Topics The importance of certain topics has changed over the decades, as depicted in Figure 2.30a. Some topics have become more important, others have declined, or stayed relatively the same. We define the importance of a topic for a decade of songs as follows: first, the LDA topic model trained on the full corpus gives the probability of the topic for each song separately. We then average these song-wise probabilities over all songs of the decade. For each of the cases of growing, diminishing and constant importance, we display two topics. The topics War and Death have appreciated in importance over time. This is partially caused by the rise of Heavy Metal in the beginning of the 1970s, as the vocabulary of the Death topic is very typical for the genre (see for instance the "Metal top 100 words" in [176]). We measure a decline in the importance of the topics Love and Family. The topics Money and Religion seem to be evergreens as their importance stayed rather constant over time.

Changes in Explicitness We find that newer songs are more likely being tagged as having explicit content lyrics. Figure 2.30b shows our estimates of explicitness per decade, the ratio of songs in the decade tagged as explicit to all songs of the decade. Note that the Parental Advisory Label was first distributed in 1985 and many older songs may not have been labelled retroactively. The depicted evolution of explicitness may therefore overestimate the "true explicitness" of newer music and underestimate it for music before 1985.

Changes in Emotion We estimate the emotion of songs in a decade as the average valence and arousal of songs of that decade. We find songs to decrease both in valence and arousal over time. This decrease in positivity (valence) is in line with the diminishment of positively connotated topics such as Love and Family and the appreciation of topics with a more negative connotation such as War and Death.

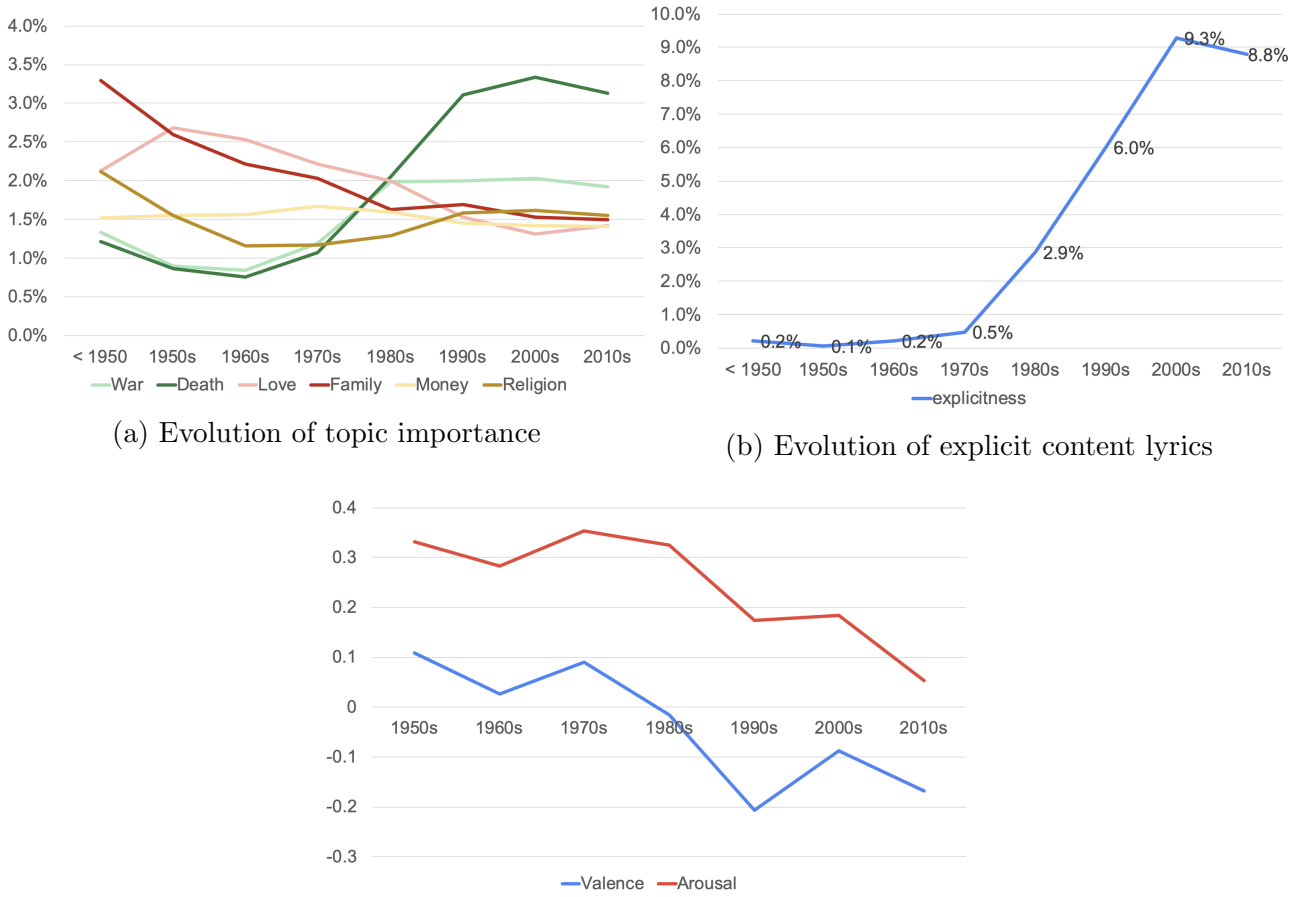
2.6.8 Related Work on lyrics processing

Detection of lyrics structure. Besides the work of [473] that we have discussed in detail in Section 2.4, only a few papers in the literature have focused on the automated detection of the structure of lyrics. [296] report experiments on the use of standard NLP tools for the analysis of music lyrics. Among the tasks they address, for structure extraction they focus on lyrics having a clearly recognizable structure (which is not always the case) divided into segments. Such segments are weighted following the results given by descriptors used (as full length text, relative position of a segment in the song, segment similarity), and then tagged with a label describing them (e.g., chorus, verses). They test the segmentation algorithm on a small dataset of 30 lyrics, 6 for each language (English, French, German, Spanish and Italian), which had previously been manually segmented.

More recently, [27] describe a semantics-driven approach to the automatic segmentation of song lyrics, and mainly focus on pop/rock music. Their goal is not to label a set of lines in a given way (e.g., verse, chorus), but rather identifying recurrent as well as non-recurrent groups of lines. They propose a rule-based method to estimate such structure labels of segmented lyrics, while in our approach we apply machine learning methods to unsegmented lyrics.

[106] propose a new method for enhancing the accuracy of audio segmentation. They derive the semantic structure of songs by lyrics processing to improve the structure labeling of the estimated audio segments. With the goal of identifying repeated musical parts in music audio signals to estimate music structure boundaries (lyrics are not considered), [118] propose to feed Convolutional Neural Networks with the square-sub-matrices centered on the main diagonals of several SSMs, each one representing a different audio descriptor, building their work on [183].

For a different task than ours, [318] use a corpus of 100 lyrics synchronized to an audio representation with information on musical key and note progression to detect emotion. Their



(a) Evolution of topic importance

(b) Evolution of explicit content lyrics

(c) Evolution of emotion

Figure 2.30: Evolution of different annotations during the decades

classification results using both modalities, textual and audio features, are significantly improved compared to a single modality.

As an alternative to our CNN-based approach, Recurrent Neural Networks can also be applied to lyrics segmentation, e.g., in the form of a sequence labeller [293] or a generic text segmentation model [275].

Text summarization. In the literature, there are two different families of approaches for automatic text summarization: extraction and abstraction [7]. *Extractive summarization methods* identify important elements of the text and generate them verbatim (they depend only on extraction of sentences or words from the original text). In contrast, *abstractive summarization methods* interpret and examine the text to generate a new shorter text that conveys the most critical information from the original text. Even though summaries created by humans are usually not extractive, most of the summarization research has focused on extractive methods. Purely extractive summaries often give better results [339], due to the fact that latter methods cope with more complex problems such as semantic representation, inference and natural language generation. Existing abstractive summarizers often rely on an extractive pre-processing component to produce the abstract of the text [49, 250]. Consequently, in this work we focus on extractive summarization methods, also given the fact that lyrics *i)* strongly use figurative language which makes abstractive summarization even more challenging; and *ii)* the choice of the words by the composer may also have an importance for capturing the style of the song.

In the following, we focus on *unsupervised* methods for text summarization, the ones targeted in our study (no available gold-standard of human-produced summaries of song texts

exists). Most methods have in common the process for summary generation: given a text, the importance of each sentence of that text is determined. Then, the sentences with highest importance are selected to form a summary. The ways different summarizers determine the importance of each sentence may differ: *Statistics-based summarizers* extract indicator features from each sentence, e.g., [169] use among others the sentence position and length and named entities as features. *Topic-based summarizers* aim to represent each sentence by its underlying topics. For instance, [221] apply Probabilistic Latent Semantic Analysis, while Latent Dirichlet Allocation is used in [16] to model each sentence's distribution over latent topics. Another type of summarization methods is *graph-based summarizers*. Three of the most popular graph-based summarizers are TextRank [319], LexRank [163], and [365]. These methods work by constructing a graph whose nodes are sentences and whose graph edge weights are sentence similarities. Then, the sentences that are central to the graph are found by computing the PageRank [358]. Contrarily to all previously described methods, systems using *supervised machine learning* form another type of summarizers. For instance, [168] treats extractive summarization as a binary classification task, where they extract indicator features from sentences of gold summaries and learn to detect the sentences that should be included in a summary.

If specific knowledge about the application scenario or the domain of the summarized text is available, generic summarization methods can be adapted to take into account the prior information. In query-based summarization [355, 467], the user's query is taken into account when generating a summary. Summarization of a scientific paper can be improved by considering the citations of it, as in [140]. However, to the best of our knowledge no summarization methods have been proposed for the domain of song texts. In this work we present a summarization method that uses prior knowledge about the text it summarizes to help generic summarizers generate better summaries.

Summaries should *i)* contain the most important information from input documents, *ii)* not contain redundant information, *iii)* be readable, hence they should be grammatical and coherent [366]. While a multitude of methods to identify important sentences has been described above, several approaches aim to make summaries less redundant and more coherent. The simplest way to evaluate summaries is to let humans assess the quality, but this is extremely expensive. The factors that humans must consider when giving scores to each candidate summary are grammaticality, non redundancy, integration of most important pieces of information, structure and coherence [400]. The more common way is to let humans generate possibly multiple summaries for a text and then automatically assess how close a machine-made summary is to the human gold summaries computing ROUGE scores [276], which boils down to measuring n-gram overlaps between gold summaries and automatic summary. More recently there have been attempts to rate summaries automatically without the need for gold summaries [342]. The key idea is that a summary should be similar to the original text in regard to characteristic criteria as the word distribution. [294] find that topic words are a suitable metric to automatically evaluate micro blog summaries.

Audio summarization. Lyrics are texts that accompany music. Therefore, it is worthwhile to see if methods in audio summarization can be transferred to lyrics summarization. In audio summarization the goal is to find the most representative parts in a song, in Pop songs those are usually the chorus and the bridge, in instrumental music the main theme. The task of creating short audio summaries is also known as audio thumbnailing [35, 103, 273], as the goal is to produce a short representation of the music that fits onto a thumbnail, but still covers the most representative parts of it. In a recent approach of audio thumbnailing [243], the authors generate a *Double Thumbnail* from a musical piece by finding the two most representative parts in it. For this, they search for candidate musical segments in an a priori unsegmented song. Candidate musical segments are defined as sequences of music that more or less exactly

repeat themselves. The representativeness of each candidate segment to the whole piece is then estimated by their fitness metric. They define the fitness of a segment as a trade-off between how exactly a part is repeated and how much of the whole piece is covered by all repetitions of that segment. Then, the audio segments along with their fitness allow them to create an audio double thumbnail consisting of the two fittest audio segments.

Songs and lyrics databases. The Million Song Dataset (MSD) project[56] [51] is a collection of audio features and metadata for a million contemporary popular music tracks. Such dataset shares some similarities with WASABI with respect to metadata extracted from Web resources (as artist names, tags, years) and audio features, even if at a smaller scale. Given that it mainly focuses on audio data, a complementary dataset providing lyrics of the Million Song dataset was released, called musiXmatch dataset[57]. It consists in a collection of song lyrics in bag-of-words (plus stemmed words), associated with MSD tracks. However, no other processing of the lyrics is done, as is the case in our work.

MusicWeb and its successor MusicLynx [8] link music artists within a Web-based application for discovering connections between them and provides a browsing experience using extra-musical relations. The project shares some ideas with WASABI, but works on the artist level, and does not perform analyses on the audio and lyrics content itself. It reuses, for example, MIR metadata from AcousticBrainz.

The WASABI project has been built on a broader scope than these projects and mixes a wider set of metadata, including ones from audio and natural language processing of lyrics. In addition, as presented in this work, it comes with a large set of Web Audio enhanced applications (multitrack player, online virtual instruments and effect, on-demand audio processing, audio player based on extracted, synchronized chords, etc.)

Companies such as Spotify, GraceNote, Pandora, or Apple Music have sophisticated private knowledge bases of songs and lyrics to feed their search and recommendation algorithms, but such data are not available (and mainly rely on audio features).

Explicit content detection. [50] consider a dataset of English lyrics to which they apply classical machine learning algorithms. The explicit labels are obtained from Soundtrack Your Brand [58]. They also experiment with adding lyrics metadata to the feature set, such as the artist name, the release year, the music energy level, and the valence/positiveness of a song. [110] apply explicit lyrics detection to Korean song texts. They also use tf-idf weighted BOW as lyrics representation and aggregate multiple decision trees via boosting and bagging to classify the lyrics for explicit content. More recently, [248] proposed a neural network method to create explicit words dictionaries automatically by weighting a vocabulary according to all words' frequencies in the explicit class vs. the clean class, accordingly. They work with a corpus of Korean lyrics.

Emotion recognition Recently, [139] address the task of multimodal music mood prediction based on the audio signal and the lyrics of a track. They propose a new model based on deep learning outperforming traditional feature engineering based approaches. Performances are evaluated on their published dataset with associated valence and arousal values which we introduced in Section 2.6.5

[482] model song texts in a low-dimensional vector space as bags of concepts, the "emotional units"; those are combinations of emotions, modifiers and negations. [486] leverage the music's emotion annotations from Allmusic which they map to a lower dimensional psychological model

[56] `http://millionsongdataset.com`

[57] `http://millionsongdataset.com/musixmatch/`

[58] `https://www.soundtrackyourbrand.com`

of emotion. They train a lyrics emotion classifier and show by qualitative interpretation of an ablated model (decision tree) that the deciding features leading to the classes are intuitively plausible. [234] aim to detect emotions in song texts based on Russell's model of mood; rendering emotions continuously in the two dimensions of arousal and valence (positive/negative). They analyze each sentence as bag of "emotional units"; they reweight sentences' emotions by both adverbial modifiers and tense and even consider progressing and adversarial valence in consecutive sentences. Additionally, singing speed is taken into account. With the fully weighted sentences, they perform clustering in the 2D plane of valence and arousal. Although the method is unsupervised at runtime, there are many parameters tuned manually by the authors in this work.

[318] render emotion detection as a multi-label classification problem, songs express intensities of six different basic emotions: anger, disgust, fear, joy, sadness, surprise. Their corpus (100 song texts) has time-aligned lyrics with information on musical key and note progression. Using Mechanical Turk they each line of song text is annotated with the six emotions. For emotion classification, they use bags of words and concepts, as musical features key and notes. Their classification results using both modalities, textual and audio features, are significantly improved compared to a single modality.

Topic Modelling Among the works addressing this task for song lyrics, [296] define five ad hoc topics (Love, Violent, Antiwar, Christian, Drugs) into which they classify their corpus of 500 song texts using supervision. Related, [176] also use supervision to find bags of genre-specific n-grams. Employing the view from the literature that BOWs define topics, the genre-specific terms can be seen as mixtures of genre-specific topics.

[287] apply the unsupervised topic model Probabilistic LSA to their ca. 40k song texts. They learn latent topics for both the lyrics corpus as well as a NYT newspaper corpus (for control) and show that the domain-specific topics slightly improve the performance in their MIR task. While their MIR task performs highly better when using acoustic features, they discover that both methods err differently. [249] apply Non-negative Matrix Factorization (NMF) to ca. 60k song texts and cluster them into 60 topics. They show the so discovered topics to be intrinsically meaningful.

[429] have worked on topic modelling of a large-scale lyrics corpus of 1M songs. They build models using Latent Dirichlet allocation with topic counts between 60 and 240 and show that the 60 topics model gives a good trade-off between topic coverage and topic redundancy. Since popular topic models such as LDA represent topics as weighted bags of words, these topics are not immediately interpretable. This gives rise to the need of an automatic labelling of topics with smaller labels. A recent approach [53] relates the topical BOWs with titles of Wikipedia articles in a two step procedure: first, candidates are generated, then ranked.

2.7 Events extraction from social media posts

Twitter has become a valuable source of timely information covering topics from every corner of the world. For this reason, NLP researchers have shown growing interest in mining knowledge from Twitter data. As a result, several approaches have been proposed to build applications over tweets, e.g., to extract structured representations/summary of newsworthy events [307, 246], or to carry out sentiment analysis to study users reactions [2, 254]. However, as we will see also in other chapters of this manuscript, processing tweets is a challenging task, since information in Twitter stream is continuously changing in real-time, while at the same time there might be a high volume of redundant messages referring to the same issue or event.

In this section, we focus on event extraction from Twitter, consisting in the automated clustering of tweets related to the same event based on relevant information such as time and

participants. Although there is no consensus in the NLP community on what an event is [421], our approach relies on the event definition by [150], i.e. "*an occurrence causing change in the volume of text data that discusses the associated topic at a specific time. This occurrence is characterized by topic and time, and often associated with entities such as people and location*".

At the time of this study, existing approaches to the task create clusters of tweets around event-related keywords [361], or NEs [307]. However, such approaches fail *i)* to capture events that do not generate spikes in the volume of tweets; and *ii)* to distinguish between events that involve the same NEs and keywords. Other approaches model the relationships between terms contained in the tweets relying on a graph representation [247], and retain the nodes with the highest number of edges as event candidates. However, the main drawbacks of these approaches are that *i)* they generate highly dense graphs, and *ii)* trending terms not related to events may be considered as event candidates.

To address such limitations, in this work we propose an unsupervised approach to detect open-domain events on Twitter, where the stream of tweets is represented through temporal event graphs, modeling the relations between NEs and the terms that surround their mentions in the tweets.

2.7.1 Approach description

In this section, we describe our approach for detecting open-domain events on tweets. The pipeline consists of the following components: Tweet pre-processing, Named Entity recognition and linking, graph creation, graph partitioning, event detection and event merging. Each step is described in the following paragraphs.

Tweet Preprocessing. The workflow starts by collecting tweets published during a fixed time window, which can be set as input parameter (e.g., 1 hour). Then, we apply common text preprocessing routines to clean the input tweets. We use TweetMotifs [350], a specific tokenizer for tweets, which treats hashtags, user mentions and emoticons as single tokens. Then, we remove the retweets, URLs, non ASCII characters and emoticons. It is worth mentioning that at this stage we do not perform stop word removal since stop words can be part of NEs (e.g., United States of America). As for hashtags, we define a set of hand-crafted rules to segment them into terms, is possible. We also try to correct misspelled terms using SymSpell[59], which matches misspelled tokens with Wordnet synsets [177].

Named Entity Recognition and Linking. We use NERD-ML [452], a Twitter specific Named Entity Recognizer (NER) tool, to extract NE mentions in the tweets, since [141] showed that it is one of the best performing tools for NER on Twitter data. Besides, NERD-ML not only recognizes the most common entity types (i.e. Person, Organization and Location), but tries also to link any term listed in external knowledge bases such as DBpedia[60] or Wikipedia. These are then associated with semantic classes in the NERD ontology.

Graph Generation. Previous works using graph-based methods to model relations between terms in text considered all terms in the input document as nodes and used their position in text to set edges [12, 484]. Such approaches may generate a dense graph, which generally requires high computational costs to be processed.

In this work, we assume that the terms surrounding the mention of a NE in a tweet define its context [349]. Thus, we rely on the NE context to create event graphs, built as follows:

[59] `https://github.com/wolfgarbe/symspell`
[60] `http://dbpedia.com/`

- **Nodes**: We consider NE and k terms that precede and follow their mention in a tweet as nodes, where $k > 1$ is the number of terms surrounding a NE to consider while building the NE context.

- **Edges**: Nodes in the graph are connected by an edge if they co-occur in the context of a NE.

- **Weight**: The weight of the edges is the number of co-occurrences between terms in the NE context. In addition, each edge maintains as a property the list of tweets from which the relationship is observed.

Formally, let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be a directed graph (or digraph) with a set of vertices $\mathcal{V}$ and edges $\mathcal{E}$, such that $\mathcal{E} \subset \mathcal{V} \bigtimes \mathcal{V}$. For any $\mathcal{V}_i \in \mathcal{V}$, let $In(\mathcal{V}_i)$ be the set of vertices that point to $\mathcal{V}_i$ (i.e. predecessors), and $Out(\mathcal{V}_i)$ be the set of vertices that $\mathcal{V}_i$ points to (i.e. successors).

Let $\mathcal{E}_i = (\mathcal{V}_j, \mathcal{V}_k)$ be an edge that connects node $\mathcal{V}_j$ to $\mathcal{V}_k$, we define ω_{ij} as the weight of $\mathcal{E}_i$, which is represented by the number of times relationships between $\mathcal{V}_j$ and $\mathcal{V}_k$ is observed in tweets published during a time window. An example of the graph created on 2011-07-07 with tweets related to the famine in Somalia and space shuttle to Mars is shown in Figure 2.31.

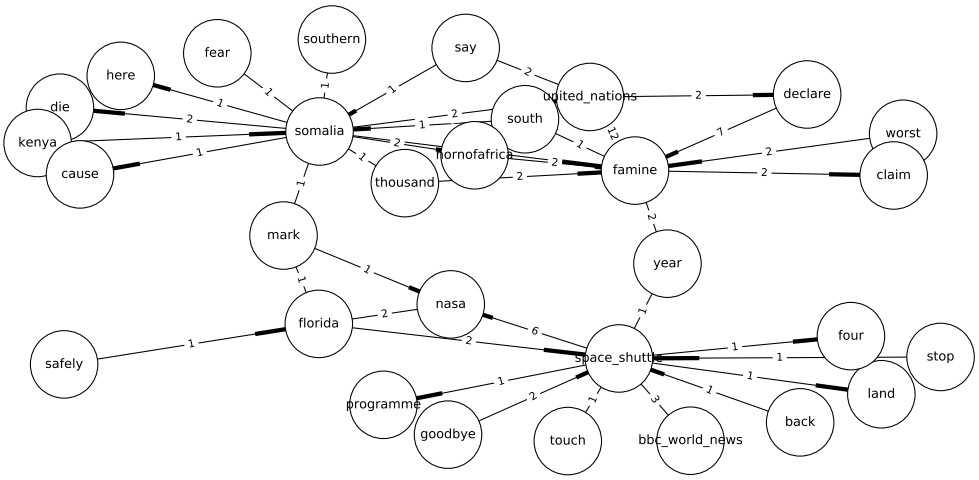


Figure 2.31: Graph generated on day "2011-07-07" from a sample of tweets related to the events about the famine in Somalia and the space shuttle to Mars.

Graph Partitioning. At this stage, an event graph is generated to model relationships between terms in the NE contexts. We apply graph theory to partition the graph into sub-graphs, which will be considered as event candidates. Tweets related to the same events usually share a few common keywords [307]. In the event graphs, this phenomenon is expressed by stronger links between nodes related to the same event. In other words, the weight of edges that connect terms from tweets related to similar events are higher than edges between nodes that connect terms from tweets related to different events. The graph partitioning purpose is to identify such edges that, if removed, will split the large graph $\mathcal{G}$ into sub-graphs.

Let $\mathcal{E} = \{(\mathcal{V}_1, \mathcal{W}_1), (\mathcal{V}_2, \mathcal{W}_2), ..., (\mathcal{V}_n, \mathcal{W}_n)\}$ be a set of pair of vertices in a strongly connected graph $\mathcal{G}$. We define λ as the least number of edges whose deletion from $\mathcal{G}$ would split $\mathcal{G}$ into connected sub-graphs. Similarly, we define the edge-connectivity $\lambda(\mathcal{G})$ of $\mathcal{G}$ of an edge set $\mathcal{S} \subset \mathcal{E}$

as the least cardinality $|\mathcal{S}|$ such that $\mathcal{G} - \mathcal{S}$ is no longer strongly connected. For instance, given the graph in Figure 2.31 as input, the deletion of edges "mark/somalia" and "year/famine" will create two strongly connected sub-graphs, where the first one contains keywords related to "famine in Somalia" and other contains keywords related to "The space shuttle to Mars".

Event Detection. We assume that events from different sub-graphs are not related to each other. Thus, in the event detection sub-module, each sub-graph is processed separately. In a study on local partitioning, [12] show that a good partition of a graph can be obtained by separating high-ranked vertices from low-ranked ones, if the nodes in the graph have distinguishable values. We use a PageRank-like algorithm [72] to rank vertices in the event-graph as follows :

$$S(V_i) = ((1-d) + d \sum_{v_j \in In(V_i)} \frac{w_{ji}}{\sum_{v_k \in Out(V_k)} \omega_{jk}} S(V_j))\epsilon_i \tag{2.4}$$

where ω_{ij} is the weight of edge connecting V_i to V_j, d a dumping factor usually set to 0.85 [72] and ϵ_i a penalization parameter for node i. In previous approaches [319], the penalization parameter is considered as a uniform distribution; instead, we define the penalization parameter of a node according to its tf-idf score. Due to redundant information in tweets, the score of the nodes can be biased by the trending terms in different time windows. Thus, we use the tf-idf score to reduce the impact of trending terms in the collection of tweets. Before computing the score with equation 2.4, we assign an initial value $\tau = 1/n$ to each vertex in the graph, where n is the total number of nodes in the graph. Then, for each node, the computation iterates until the desired degree of convergence is reached. The degree of convergence of a node can be obtained by computing the difference between the score at the current iteration and at the previous iteration, which we set to 0.0001 [72].

As shown in Algorithm 1, we start by splitting the vertex set into high-ranked and low-ranked vertices based on a gauged parameter α (Line 3). Next, we process the vertices in the high-ranked subset starting from the highest ones, and for each candidate we select the highest weighted predecessors and successors as keywords for event candidates (Lines 4-9). After removing the edges between the keywords from the graph, if it becomes disconnected, we also consider the disconnected nodes as keywords for the event candidate (Lines 10-13). Based on the semantic class provided by the NER tool (see Section 2.7.1), we divide the keywords related to an event in the following subsets: *what* (i.e., the type of the event), *where* (i.e., the location in which the event happens), *who* (i.e., the person or organization involved). As for the date, we select the oldest tweets that report the event.

In the second stage of Algorithm 1, we further process the event candidates. First, we merge duplicate event candidates (Lines 22-35), i.e. those sharing common terms and having the same location or participants in the considered time window. A new event is thus built from the combination of terms and entities of the two event candidates. An event is considered as valid if at least a NE is involved, and if it occurs in a minimum number of tweets provided as input parameter.

Event Merging. We consider events in different time-windows as duplicate if they contain the same keywords, entities (e.g., person, organization, location) in an interval of k days, where k is an input parameter. When a new event is found as duplicate, we merge it with the previous detected event.

Algorithm 1 Algorithm to process a given event-graph to retrieve important sub-events.

```
function GRAPH_PROCESSING(G, α)
    E = ∅
    H = {v_i ∈ vertex(G) if score(v_i) >= α}                ▷ Equation 2.4
    while H ≠ ∅ do
        G' = G.copy()
        v_i = H.pop()
        p = max(W_j ∈ In(v_i)
        s = max(W_j ∈ Out(v_i)
        keywords = set(p, v_i, s)
        G'.remove_edges((p, v_i), (v_i, s))
        if not G'.connected() then
            append(keywords, disc_vertices(G'))
        end if
        who = person||organization ∈ keywords
        where = location ∈ keywords
        what = keywords − who − where
        tweets = tweet_from(keywords)
        when = oldest(tweets, date)
        event =< what, who, where, when >
        append(E, event)
    end while
    for e ∈ E do
        for e' in E do
            if what(e) ∩ what(e') then
                if who(e) ∩ who(e') then
                    merge(e, e')
                end if
                if where(e) ∩ where(e') then
                    merge(e, e')
                end if
            end if
        end for
        if not who(e) or not where(e) then
            discard(E, e)
        end if
    end for
    return E
end function
```

2.7.2 Experiments

Given a set of tweets, our goal is to cluster such tweets so that each cluster corresponds to a fine-grained event such as "Death of Amy Winehouse" or "Presidential debate between Obama and Romney during the US presidential election". We first describe the datasets, then we present the experimental setting. This section ends with a comparison of the obtained experimental results with state-of-the-art approaches.

Dataset. We test our approach on two gold standard corpora: the First Story Detection (FSD) corpus [377] and the EVENT2012 corpus [308].

FSD. The corpus was collected from the Twitter streaming API[61] between 7th July and 12th September 2011. Human annotators annotated 3,035 tweets as related to 27 major events occurred in that period. After removing tweets that are no more available, we are left with 2,342 tweets related to one out of the 27 events. To reproduce the same dataset used by other state-of-the-art approaches, we consider only those events mentioned in more than 15 tweets. Thus, the final dataset contains 2,295 tweets describing 20 events.

EVENT2012. A corpus of 120 million tweets collected from October to November 2012 from the Twitter streaming API, of which 159,952 tweets were labeled as event-related. 506 event types were gathered from the Wikipedia Current Event Portal, and Amazon Mechanical Turk was used to annotate each tweet with one of such event types. After removing tweets that

[61] `dev.twitter.com/streaming/overview`

are no longer available, our final dataset contains ∼43 million tweets from which 152,758 are related to events.

Experimental Setting. For each dataset, we compare our approach with state-of-the-art approaches. For the FSD dataset, we compare with LEM Bayesian model [496] and DPEMM Bayesian model enriched with word embeddings [497]. For the EVENT2012 dataset, we compare our results with Named Entity-Based Event Detection approach (NEED) [307] and Event Detection Onset (EDO) [247].

In order to simulate a real scenario where tweets are continuously added to a stream, we simulate the Twitter stream with a client-server architecture which pushes tweets according to their creation date. We evaluate our approach in two different scenarios: in the first scenario, we consider tweets from the FSD dataset that are related to events and we classify them into fine-grained event clusters. In the second scenario, we adopt a more realistic approach in that we consider all the tweets from the EVENT2012 dataset (i.e., event-related and not event-related ones), and we classify them into event clusters, discarding those that are not related to events. Our approach requires a few parameters to be provided as input. In the experiments reported here, we process the input stream with fixed time-window $w = 1$ hour. The minimum number of tweets for event candidates is set to $n = 5$. Finally, we empirically choose $t = 3$ days as the interval of validity for the detected events.

Results. Performance is evaluated both in terms of P/R/F1 and cluster purity.
Results on the FSD dataset: In this scenario, we consider an event as correctly classified if *all* the tweets in that cluster belong to the same event in the gold standard, otherwise the event is considered as misclassified. Due to the low number of tweets, we set the gauged parameter $\alpha = 0.5$ as the minimum score for nodes in the graph to be considered as useful for events. Table 2.19 shows the experimental results yielded by our approach in comparison to state-of-the-art approaches. Our approach outperforms the others, improving the F-score by 0.07 points w.r.t. DPEMM and by 0.13 w.r.t. LEM.

Approach	Precision	Recall	F-measure
LEM	0.792	0.850	0.820
DPEMM	0.862	0.900	0.880
Our Approach	0.950	0.950	0.950

Table 2.19: Evaluation results on the FSD dataset.

Furthermore, we evaluate the quality of the events, i.e. the clusters, in terms of purity, where the purity of an event is based on the number of tweets correctly classified in the cluster and the number of misclassified tweets. More specifically, purity is computed as: $P_e = \frac{n_e}{n}$, where n_e is the number of tweets correctly classified and n the total number of tweets classified in that cluster. Figure 2.32 reports the purity of our approach compared to LEM and DPEMM, where each point (x, y) denotes the percentage of events having purity less than x. It can be observed that 5% of the events detected as well as DPEMM have purity less than 0.65 compared to 25% for LEM, while 95% of the events detected have purity higher than 0.95 compared to 75% for DPEMM and 55% for LEM.
Results on the EVENT2012 dataset: We also evaluate our approach on the EVENT2012 dataset using a more realistic scenario in which all the tweets (i.e., events related and non-event related tweets) are considered. Compared to the FSD dataset, the EVENT2012 dataset has more events and tweets and thus a larger vocabulary. We set the cutting parameter $\alpha = 0.75$ as the minimum score of nodes in the graph to be considered as important for events. We further

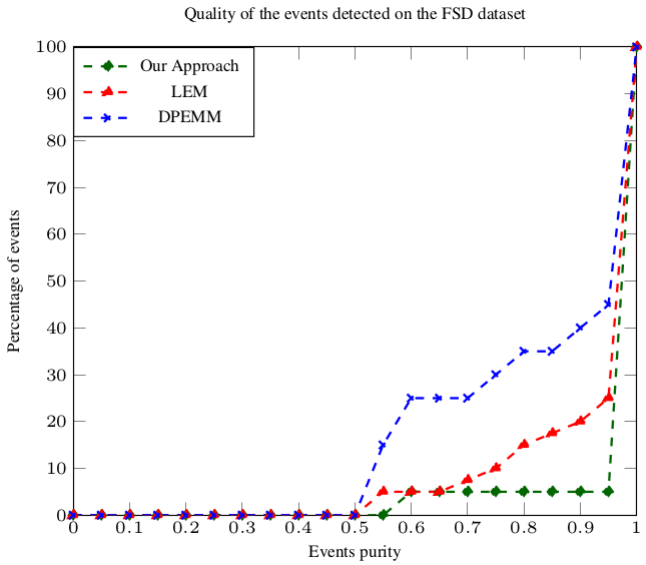


Figure 2.32: Purity of the events detected by our approach, LEM and DPEMM on the FSD dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.

detail the importance of the parameters α later in this section. Also, since we include both event-related and not event-related tweets, we consider an event as correct if 80% of the tweets belong to the same event in the ground truth. Table 2.20 reports on the experimental results compared to the NEED and EDO approaches. In general, our approach improves the f-score by 0.07 points w.r.t. EDO and 0.23 points w.r.t. NEED. After a manual check of the output, we noticed that some issues with precision may depend on the quality of the dataset, since some tweets related to events were not annotated as such in the gold standard. For example, we found that 9,010 tweets related to "BET hip hop award" were not annotated. The same was found for tweets concerning large events such as "the Presidential debate between Obama and Romney" or the "shooting of Malala Yousafzai, the 14-year old activist for human rights in Pakistan".

We also evaluate the purity of the events detected by our approach (Figure 2.33). We can observe that the quality of the detected events is lower than for the events detected on the FSD dataset. For instance, more than 20% of the detected events have purity *lower* than 0.7. As expected, event purity is mainly affected by the inclusion in the clusters of non event-related tweets.

Approach	Precision	Recall	F-measure
NEED	0.636	0.383	0.478
EDO	0.754	0.512	0.638
Our Approach	0,750	0.668	0.710

Table 2.20: Evaluation results on the EVENT2012 dataset.

Effect of the Cutting Parameter. We further experiment on the impact of the dangling parameter on the output of our model. The dangling parameter α is used to separate the nodes of the event graph into high-ranked and low-ranked nodes, where the high-ranked nodes are used to extract keywords related to event candidates. We experiment different values for "α" and we evaluate their impact on the performance of our approach on both datasets.

In Figure 2.34a we show the performance of our model for $0 < \alpha \leq 4$ on the FSD dataset. We observe that higher value of α gives higher precision while lowering the recall. More specifically,

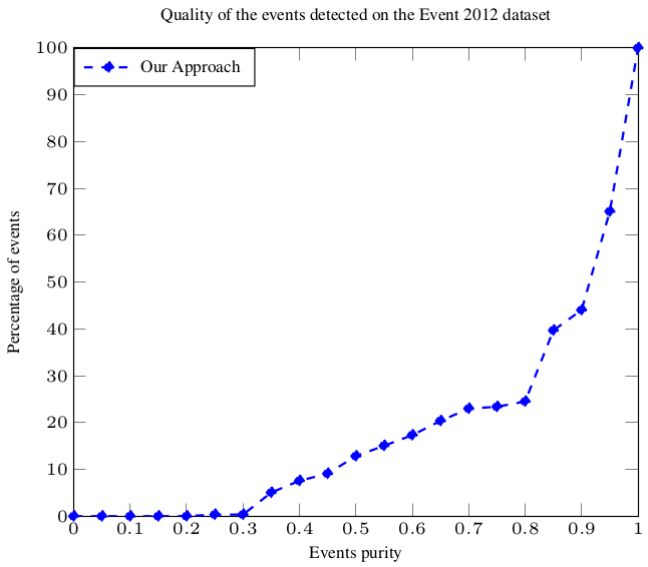
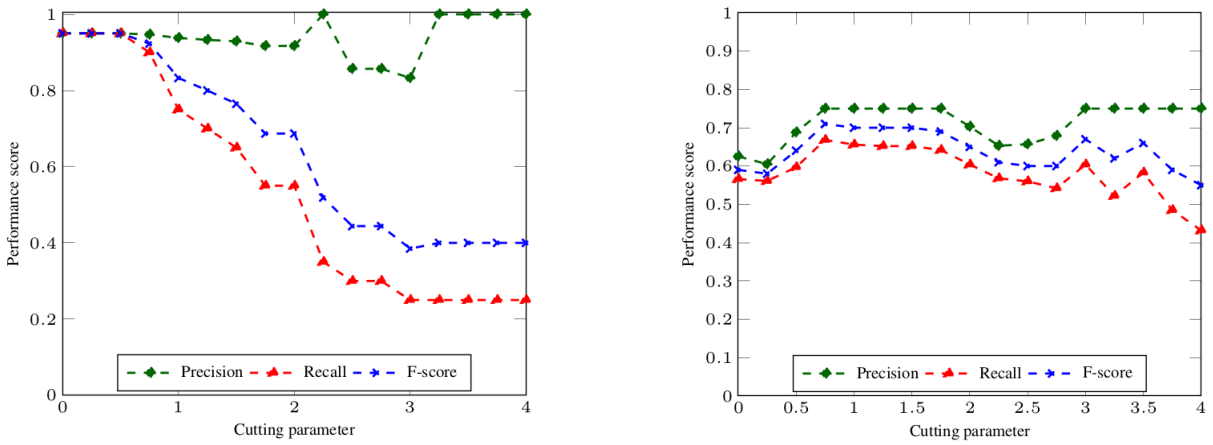


Figure 2.33: Purity of the events detected by our approach on the event 2012 dataset. The y-axis denotes the percentage of events and the x-axis the purity of the events.



(a) Effect on FSD

(b) Effect on EVENT2012

Figure 2.34: Effect of the cutting parameter (α) on the performance of our approach.

for $\alpha \geq 3$ we obtain 100% precision and recall lower than 50%. On the other hand, the best performance is obtained for $\alpha \leq 0.5$. Since the FSD dataset contains $\sim 6,000$ unique words, at each time window the generated graph is strongly connected, thus the average minimum score of the nodes is higher than 0.5. For values higher than 0.5, important terms referring to events are ignored, mainly when they are related to events that do not generate a high volume of tweets. In our experiments, we also observe that higher values of α mostly affect the recognition of events with low number of tweets.

Figure 2.34b shows the performance of our model for different values of α on the EVENT2012 dataset. We observe that for different values of α, both precision and recall are affected. More specifically, the recall of the model tends to decrease for lower values of α. Without edge cutting (i.e. $\alpha = 0$), the recall of our model is similar to EDO. Overall, the impact of α is bigger on the EVENT2012 dataset than on FSD dataset. The variation of precision and recall curves is smaller for consecutive values of α w.r.t. FSD. There are two main reasons for that: *i)* the EVENT2012 dataset has a richer vocabulary, and *ii)* many events in the EVENT2012 dataset are similar to each other.

2.8 Building events timelines from microblog posts

In the following we present another work on events extraction, to build events timelines from microblog posts. We focus in particular on sport events timelines. Historically, sports fans have watched matches either at the stadium or on TV, or have listened to them on the radio. In the latest years, however, social media platforms have become a new communication channel also to share information and comment on sports events, thus creating online communities of sports fans. Microblogs are particularly suitable, thanks to their coverage and speed, making them a successful channel to follow and comment on events in real time. Also sports teams and medias have benefited from these platforms to extend their contact networks, increase their popularity and exchange information with fans [193, 356]. The need to monitor and organize such information is particularly relevant during big events like the Olympic Games or FIFA World Cup: several matches take place in a limited time span, sometimes in parallel, and summaries are manually made by journalists. A few approaches have recently tried to automatize this task by recognizing actions in multimedia data [212, 416, 415].

In this work, we investigate whether the same task can be performed relying only on user-generated content from microblogs. In fact, opinions shared by fans during sports matches are usually reactions to what is happening in the game, implicitly conveying information on the ongoing events. Existing works aimed at building complete summaries of sports games from tweets [343, 484] rely on the observation of peaks in the tweets' volume. Even though such approaches effectively detect the most salient actions in games (e.g., goals), they fail to capture actions that are not reported by many users (e.g., shoots). Moreover, they focus only on specific information: for example, [286] and [9] are respectively interested in detecting in soccer games goals, yellow and red cards, and in detecting time and keywords, ignoring the players involved in the actions.

In this work we perform a more complex task: we create a fine-grained, real-time summary of the sub-events occurring in sports games using tweets. We define a sub-event in a match as an action that involves one or many participants (e.g., a player, a team) at a given time, as proposed by [150]. More specifically, we want to address the following research questions: *i) Is it possible to build detailed sports games summaries in a unsupervised fashion, relying only on a controlled vocabulary?*, and *ii) To what extent can Twitter be used to build a complete timeline of a game? Is information retrieved via Twitter reliable and sufficient?*

2.8.1 Proposed approach

Although the approach we propose to detect sub-events in sports games and to build a timeline (Figure 2.35) is general-purpose, we take as an example soccer games, so that we can use a consistent terminology. The pipeline can be applied to any sports as long as it is represented in the Sports Markup Language [128].

First, a module for information extraction identifies actions (e.g., goals, penalties) and participants (e.g., player's names, teams) mentioned in tweets, setting relations between them (see examples in Table 2.21). Then, participants, actions and relations are modeled together in a temporal event-graph, taking into account also the time of the tweet. This leads to the creation of a timeline where actions and participants are connected and temporally ordered. The modules of this pipeline are described in detail in the following sections.

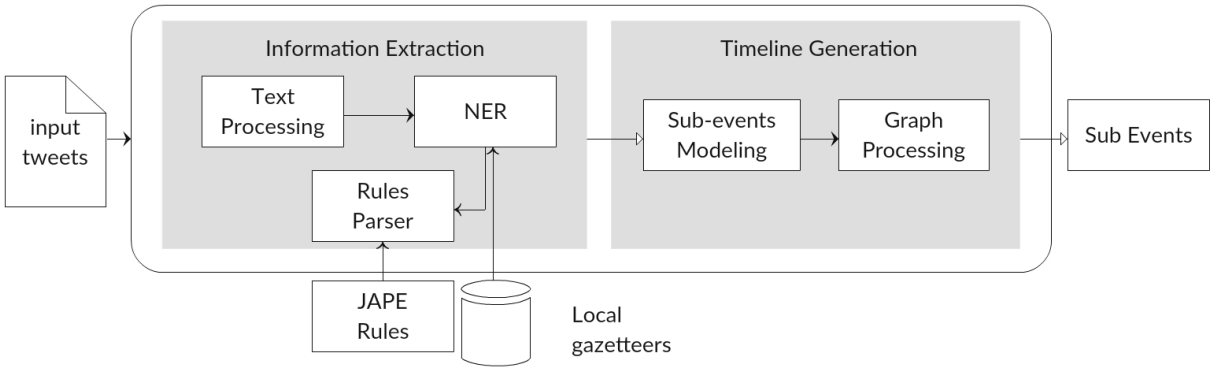


Figure 2.35: Sub-events extraction pipeline.

Tweets	Action	Particip.
kick off.... #engwal #euro2016 #teamengland	D1P	england wales
how has ramsey not got a yellow card yet every attempt to tackle has been a foul.	CJA	ramsey wales
goaaaaaaaaaaal from bale woah #eng 0-1 #wal	BUT	bale wales

Table 2.21: Detected actions and participants in tweets (England-Wales, June 16, 2016. D1P: First period begins, CJA: Yellow card, BUT: Goal).

2.8.2 Information Extraction

The first module retrieves participants and sub-events (or actions)[62] from tweets, and sets relations between them. In the case of soccer, actions are defined by FIFA, e.g., goals, penalties, yellow/red cards, etc. Participants are the actors who induce the actions. For soccer games, they are players and teams. To extract information we use GATE [130], because it includes a highly flexible NEs Recognition (NER) tool that allows the integration of custom gazetteers. To detect *actions*, we update its gazetteer based on the Sports Markup Language, a controlled vocabulary used to describe sports events. SportsML core schema provides concepts allowing the description of events for 11 major sports including Soccer, American football, Basketball and Tennis. For soccer games, we extract actions such as goals, substitutions, yellow/red

[62]In this work we use interchangeably the terms actions and sub-events to refer to actions in a sports game.

cards and penalties. Furthermore, we enrich the list of actions with synonyms extracted from Wordnet [177].

As for *participants*, we update the gazetteer using the football-data API[63] that, given a soccer game in input, returns the name of the teams and their players. We also apply some heuristics so as to associate different spelling variations to players' and teams' names. This is done by considering separately or by combining the different parts of the players' names (i.e. first name and last-name). For instance, *"giroud"*, *"oliviergiroud"* or *"olivier_giroud"* are all associated with *"Olivier Giroud"*, a player in the French national team.

We first pre-process the data using GATE in-built tweet normalizer, tokenizer and PoS-tagger. Then, we use the NER module integrating the two custom gazetteers we created. We also set links representing relations between actions and participants by means of JAPE (Java Annotation Pattern Engine) rules, a GATE-specific format to define regular expressions needed for pattern matching. Since relations detected through JAPE rules tend to be very accurate, we assign a weight = 2 to edges extracted from such rules. If an action and a participant appear in the same tweet but are not matched through a JAPE rule, we set a link with weight = 1, to account for a lower precision.

2.8.3 Timeline creation

Modeling sub-events. The output of the information extraction module (Figure 2.35) is a list of tuples $\langle a, p, t, \omega \rangle$, where a is a sports action, t the timestamp of the tweet and p the participant involved and ω is the weight of the edge connecting a and p. These tuples are used to build a temporal event graph (see Figure 2.36). To retain temporal information on the sub-events, we split the game in fixed time windows, and create an event-graph that models the relationships between actions and participants for each time window. We refer to such graphs as *temporal graphs* [454] and we build them as follows:

- *Nodes*: Actions and participants are represented by nodes in the event-graph. First, we retrieve the nodes of the actions, and then we add the connected participants' nodes;

- *Edges*: Nodes are connected by an edge if a relation can be set in the tweets published during the time-window. The occurrence of this relation is used to increase the weight of the edges. Relationships between participants are created for actions involving 2 or more participants (e.g., a substitution).

Fig. 2.36 shows a temporal graph at time-window 22 of the game between England and Wales (Game #16 on June 16, 2016): we observe edges linking participants, e.g., connecting the node "Sterling" and "Vardy", retrieved from tweets requesting the substitution of "Sterling" by "Vardy". Both are linked also to the node "England", i.e. their team.

Processing the event-graphs. At this stage, the weighted relations between actions and participants are considered as sub-event candidates. We cannot automatically include them in the timeline because they could represent opinions or wishes of the fans, as: *"how has ramsey not got a yellow card yet every attempt to tackle has been a foul"*. In general, we may assume that real sub-events in a game are reported by many users, while an action reported by a few users only is more likely to be a subjective post reflecting an opinion.

Most of the existing work set an empirical threshold to measure the importance of the actions [9, 299]. However, we observe that the number of tweets generated for a given action is highly dependent on the game and the team or players involved. Thus, we find it useful to

[63] `http://api.football-data.org`

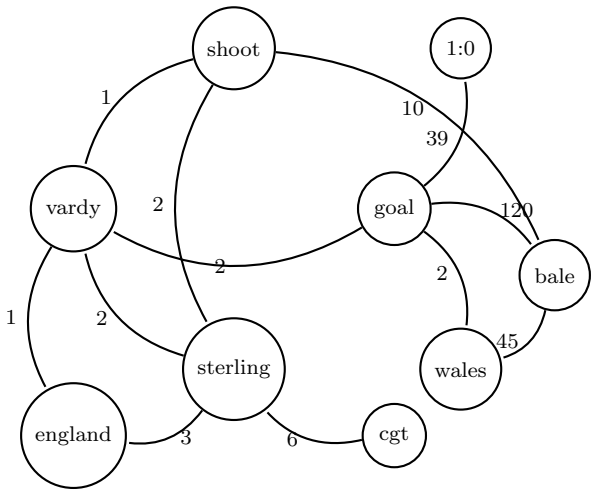


Figure 2.36: Event-graph for England-Wales.

tune the thresholds by taking into account both the type of the action and the popularity of the teams involved in the game.

For each action belonging to a certain sport, we manually define an empirical threshold according to the importance of the action. For soccer, we can assume that a goal will trigger a higher number of tweets than a shoot. These empirical values can be defined by domain experts for each category of the sports we want to track. Based on the predefined thresholds, the interest of the games for people and the popularity of the opponent teams, we adjust the empirical thresholds using Kreyszig standard score formula [256] as follows :

$$\varphi_{a,t} = \epsilon_a * \frac{\eta_{g,t} - \overline{\eta}_g}{\sigma_g} \tag{2.5}$$

where $\varphi_{a,t}$ is the threshold for action a at time t of the game, ϵ_a the empirical threshold for a, $\eta_{g,t}$ the count of tweets related to the game at time t, $\overline{\eta_g}$ the mean count, σ_g the standard deviation of tweets related to the game in the past time windows.

Ranking Sports Actions. Let $A = \langle a, p, t, \omega \rangle$ be a quadruplet modeling an action a at time t, involving participants p and weighted by ω (i.e. the number of edges connecting a and p in the event graph). For each participant, we compute a standard score as follows:

$$z_{a,p,t} = \frac{\eta_{\omega_i} - \overline{\eta}_{\omega_i}}{\sigma_{\omega_i}} \tag{2.6}$$

where η_ω is the weight of the edge in graph G that connects nodes a and p, $\overline{\eta}_\omega$ is the mean count of all the actions of type a induced by p, and σ_ω is the standard deviation of relationship between a and p over all past time windows. Thus, we evaluate the action by taking the ratio between the standard score for each participant and the total standard scores for all the participants as follows :

$$z_{a,t} = \frac{z_{a,p_i,t}}{\sum\limits_{p_i \in P} z_{a,p_i,t}} \tag{2.7}$$

At a given time t an action is added to the timeline iff there exists at least a participant p such that $z_{a,t} \geq \varphi_{a,t}$.

As shown in Algorithm 2, we first merge the current event graph and the graph from the previous time window (Line 1). Then, from the merged graph, we collect all vertices of type $foot\_action$ and for each we retrieve all connected nodes as participants of the action (Lines 4-6). We compute the adaptive threshold for each action and a standard score for each participant

using equation 2.5 and 2.6, respectively (Lines 7-9). Finally, sub-event candidates are created with participants that have a score higher than the threshold of the action (Lines 10-16). For some actions, participants may not be required (e.g., beginning/end of periods in soccer), for such actions we consider both teams as participants in order to comply with equations (2.6 and 2.7). We remove from the event graph actions and participants involved in sub-events. Besides, nodes that were not related to sub-events are kept to be processed in the next time-window. However, if a node cannot be confirmed as related to sub-events in two consecutive time windows, we consider it as noise and simply discard it.

Before putting sub-events on a timeline, we perform a final check to see whether they have not been validated in the previous time window. If yes, it means that an action overlaps two time-windows, and the timestamp of the event must be updated, matching the time of the first occurrence. We consider two events identical if: *i)* they mention the same action and participants; *ii)* the number of tweets reporting the more recent action is lower than the number of tweets on the old one.

Algorithm 2 Algorithm to process a given event-graph to retrieve important sub-events.

```
1: function GRAPH_PROCESSING(G_t, G_{t-1}, t)   ▷ G_t - Event graph at time t, G_{t-1} - Event graph at t-1, t - current time
2:     G = merge (G_t, G_{t-1})
3:     E = ∅
4:     for vertex ∈ G.vertices() do
5:         if vertexisfoot_action then
6:             P = G.neighbors(node)
7:             a = node.action
8:             φ_{a,t} = compute (a, t)                                   ▷ equation 2.5
9:             z_{a,t} = compute (a, P, t)                                ▷ equation 2.7
10:            for z ∈ z_{a,t} do
11:                if z ≥ φ_{a,t} then
12:                    event = (a, p, t)
13:                    E append (a, p, t)
14:                    G delete (a, p)
15:                end if
16:            end for
17:        end if
18:    end for
19: end function
```

2.8.4 Experiments

Dataset. We experiment our framework on the Hackatal 2016 dataset[64], collected during the EURO 2016 Championship. A set of keywords were manually defined, including hashtags (#euro, #euro2016, #football) and the names of the teams involved in the competition (e.g., France) as well as their short names (e.g., #FRA) and hashtags related to current games (e.g., #FRAROM for the game between France and Romania). For each game, tweets were collected for a two-hour time span, starting at the beginning of the game. For comparisons and to limit the complexity of the processing pipeline, we limit our analysis to tweets in English.

The dataset also contains the summary of the salient sub-events in each game, retrieved from journalistic reports (e.g., LeFigaro[65]). We consider these summaries as the ground truth while evaluating our approach. These summaries are defined as a set of triples ⟨`time, action, participant`⟩ where "time" is the time the sub-event occurs, the "action" is the type of the sub-event and "participants" are players or teams involved in the action. The sub-events include: the beginning of the periods (F1P, D1P), end of the periods (F1P, D2P), Shoot (TIR), Goal (BUT), Substitution (CGT), Red card (CRO) and Yellow card (CJA) (see Table 2.22).

[64] `http://hackatal.github.io/2016/`.
[65] `http://sport24.lefigaro.fr`

Time	Action	Participants
15:02	D1P	–
15:09	TIR	Sterling
...	...	...
15:44	BUT	Bale
15:48	F1P	–
16:04	CGT	Sterling;Vardy
16:18	BUT	Vardy

Table 2.22: A few examples of the sub-events that occurred in the game between England and Wales.

	Loose			Partial			Complete		
actions	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
goal	0.745	0.512	0.549	0.670	0.456	0.493	0.623	0.405	0.444
card	0.758	0.560	0.622	0.693	0.506	0.568	0.600	0.433	0.516
subt	0.859	0.629	0.693	0.627	0.460	0.510	0.501	0.374	0.438
shoot	0.643	0.203	0.292	0.571	0.185	0.264	0.548	0.167	0.243
period	0.814	0.656	0.706	0.655	0.517	0.562	0.585	0.462	0.523

Table 2.23: Experimental results of our approach for 24 games in the first stage of the Euro 2016 dataset

Experimental setting. We simulate the Twitter stream by grouping the tweets related to a game in intervals of two minutes, which we refer to as *time-windows*. Thus, we collect all the tweets published in a time-window in a single document which we give in input to our algorithm. In the preprocessing phase, we remove re-tweets if the original tweet is already in the collection, and we consider one tweet per user in a time window. The input tweets are then analyzed with GATE. We use the JGraph library [340] to create the event-graph. At each time-window, we create a new graph to model the relation between actions and participants detected in tweets. We process the event-graph with Algorithm 2 to detect real sub-events found in tweets.

Evaluation strategies. We report on two different evaluation strategies. In the first one, we compare the output of our framework against the state of the art approach [9]. There, sub-events are detected by identifying spikes in the Twitter stream. Since they do not detect participants, in this first comparison we also limit our evaluation to the action timeline, letting out additional information. We also compare the results with the gold standard timeline from manually created summaries by sports journalists. We show the results for three sample matches in Figures 2.38, 2.39 and 2.40.

In the second evaluation strategy, we evaluate our approach against the gold standard data described above. This time we include also the sub-event type, the time and participants information. Also, we consider three evaluation settings, namely *complete* matching, *partial* matching and *loose* matching. In the complete matching mode, we evaluate each sub-event detected by our system by taking into account the type of the sub-event, the participants and the time. A sub-event is considered correct if all three elements are correctly identified. In the partial mode, we consider the time and the type of the sub-events; and in the loose mode, we only consider the type. We set the error margin to 2 minutes while comparing the time, since this is the duration of the time-windows used to build the temporal graphs. Table 2.23 reports P/R/F1 for the same sample matches described above, as well as an average of the scores for 24 matches in the first stage of the competition.

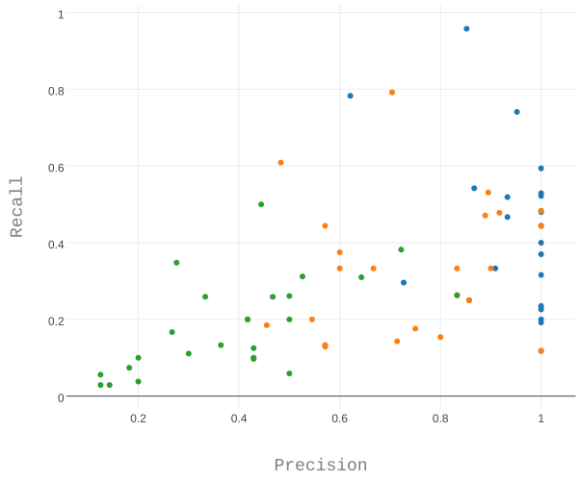


Figure 2.37: Precision Recall chart of the performances of our approach. X-axis is the average precision and Y-axis the average recall. Blue dots represent the loose matching, orange dots the partial matching and green dots the complete matching.

2.8.5 Results and discussion

The overall evaluation concerning the first 24 games in the EURO 2016 Championship (Table 2.23) shows that the approach is very accurate in some cases, while it suffers from low performance, especially recall, in other settings. If we compare the different actions (left-most columns in the table), we observe that the best performance is obtained when recognizing the start and the end of the match (last line in the table). For other actions, the performance varies across the three evaluation modes. For example, when considering participants to *shoot* actions, the approach fails to identify the correct player, probably because other players are likely to be mentioned in the same tweet. Figure 2.37 provides an overview of the obtained performances with the different evaluation strategies.

We further focus on three sample matches: we plot in Figures 2.38, 2.39 and 2.40 the sub-events detected by [9], those detected by our approach, as well as the gold standard ones. We report in Tables 2.24, 2.25 and 2.26 P/R/F1 measures for the loose, partial and complete evaluation strategy.

The first game, England - Wales gained particular attention on Twitter. Figure 2.38 shows the distribution of tweets during the game (in gray), distinguishing between tweets explicitly mentioning England (red line) and Wales (green). The blue dots correspond to the sub-events identified by [9]'s approach, while those detected by our approach and the ground truth are represented with yellow and green dots, respectively. The graphical representation shows that there is a significant correspondence between the sub-events detected by our approach and the gold standard ones. We can also observe that [9] fail to detect sub-events that do not produce spikes in the volume of tweets. Table 2.24 shows for the same match the average performance of our approach. In this case, our performance is affected by problems in detecting actions of type *substitution* and *shoots*.

Methods	Prec	Rec	F-score
loose	0.852	0.958	0.902
partial	0.630	0.708	0.667
complete	0.444	0.500	0.470

Table 2.24: Performance on England- Wales.

A second example is the match France - Romania (see Figure 2.39). Although the game

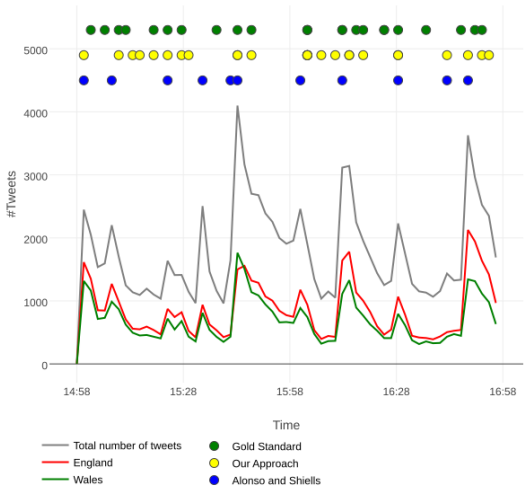


Figure 2.38: Sub-events in England-Wales.

was quite debated on Twitter, a few spikes were detected in the stream. In fact, during the first period the teams were barely mentioned, as indicated by the red and green curves on the graph. Instead, other teams were mentioned, which were not directly involved in the game. The second period seemed to be more interesting in terms of sub-events. Table 2.25 shows our performances. We obtain a 91.3% precision in the loose mode, since we detect 23 out of 34 sub-events in the game compared to 9 identified by [9], and 21 of the detected sub-events were associated to the correct actions. However, the latency between the sub-events detected by our approach compared to the ground truth contributes in decreasing the performance of our approach in both intermediate and complete matching. For example, there is a huge peak at time 22:24 when *Stancu* equalizes for Romania, but we detect this action four minutes later since most of the tweets in that time span discuss the penalty issue rather than the goal.

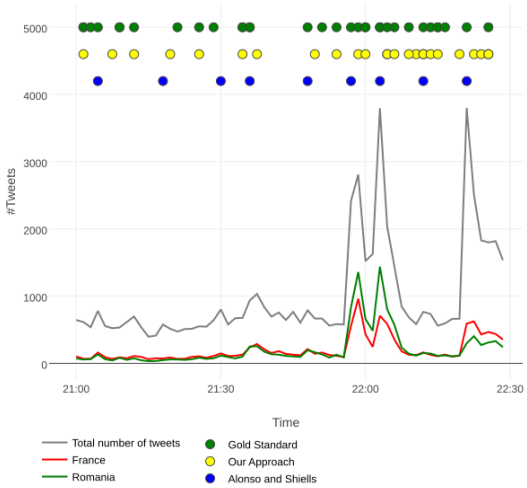


Figure 2.39: Sub-events in France-Romania.

Methods	Prec	Rec	F-score
loose	0.913	0.656	0.763
partial	0.696	0.500	0.582
complete	0.609	0.438	0.510

Table 2.25: Performance on France- Romania.

As a third example, we consider Belgium - Italy, that was less popular in terms of tweets than the previous ones. A few peaks are detected in the game (Figure 2.40). This affects negatively the number of sub-events found by [9], while our approach proves to have a better coverage, even if recall is on average lower than for the other matches. In most cases, we detect mentions of the actions, but we fail to detect the participants. Table 2.26 shows the overall performance of our approach. In the ground truth there were only a few tweets related to this game, and $\sim$ 50% of them were shoots. Our approach failed to identify them, impacting on the recall. On the other hand, all the events detected were correct.

Methods	Prec	Rec	F-score
loose	1.000	0.448	0.619
partial	0.923	0.414	0.572
complete	0.846	0.379	0.523

Table 2.26: Performance on Belgium- Italy.

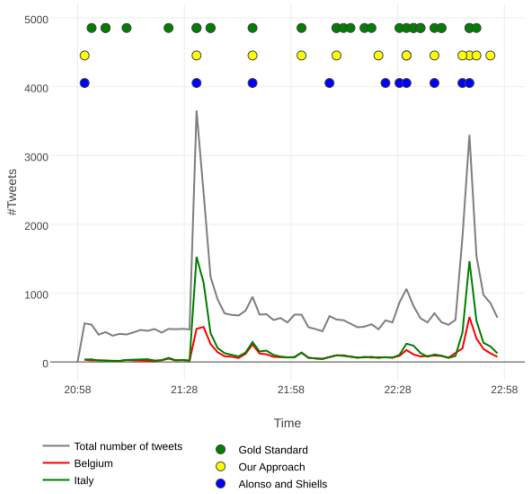


Figure 2.40: Sub-events in Belgium- Italy.

2.8.6 Related Work on events extraction

Existing approaches to extract events from tweets can be divided into two main categories, namely closed-domain and open-domain event detection systems [19]. In the closed-domain, approaches are mainly focused on extracting a particular type of event, as for instance natural disasters [359]. Works in the closed-domain scenario are usually cast as supervised classification tasks that rely on keywords to extract event-related messages from Twitter [468], to recognize event patterns [383] or to define labels for training a classifier [11, 401].

The open-domain scenario is more challenging, since it is not limited to a specific type of event and usually relies on unsupervised models. Among the works applying an unsupervised approach to event detection on Twitter, [307] create event clusters from tweets using NE mentions as central terms driving the clusters. Thus, tweets mentioning the same entities are grouped together in a single cluster. Experiments on a public dataset of tweets show that this strategy outperforms other approaches such as Latent Sensitive Hashing [376]. Similarly, [217] create clusters based on cosine similarity among tweets. Both works do not consider the temporal aspect of events and fail to capture terms or entities involved in different events at different time periods.

Very recently, [497] use a non-parametric Bayesian Mixture Model leveraged with word embeddings to create event clusters from tweets. In this approach, events are modeled as a 4-tuple $\langle y, l, k, d \rangle$ modeling non-location NEs, location NEs, event keywords and date. The work was focused on detecting events given a set of event-related tweets, which is however not applicable to a real scenario, where the stream of tweets can also contain messages that are not event-related. This scenario is simulated in the second experiment presented in this section.

Most of the works that analyze the content of tweets for tracking sport events are based on spike detection on the stream of messages, to detect sub-events. To summarize event streams, [343] propose a method that identifies spikes in Twitter feed and selects tweets from a sub-event by scoring each of them based on phrase graph [410]. This method may produce unexpected summary if most of the tweets published during the spike are not related to the sub-event. [258] generate live sports summary by prioritizing tweets published by good reporters. First, they identify spikes in the stream of an event as indicators of sub-events, and then the system tries to generate a summary by measuring the explanatory of the tweet by the presence of player's names, team names and terms related to the event. Similarly, when a spike is detected, [9] analyze the tweets published during the period to identify the most frequent terms which they use to describe spikes in a tweets' histograms (spikes are considered as sub-events). To summarize tweets on football, [239] create event clusters with similar documents, that are then automatically classified as relevant to football actions.

In the case of sports games, spikes do not necessarily characterize a sub-event. For example, when the crowd disagrees with the referees or a player, emotional tweets to express disagreement are published. On the other hand, actions with low importance (e.g., a shoot) or actions produced by non-popular teams or players (e.g., Albania) may not produce peaks in the volume of tweets. Thus, approaches solely based on spikes detection are unable to capture those actions. In our approach, we rely on Named Entities (NEs) to identify whether or not a tweet is related to a sports event. Besides, we rely on an adaptive threshold tuned according to the actions and the team (or player) of interest to evaluate whether or not the actions should be added to the timeline.

2.9 Conclusions

In this section, we have described the research contributions related to extract information from text to generate structured knowledge in different application scenarios.

In the first part of this chapter, we focused on the mining of semantic knowledge from the Web for the robotics domain, to integrate such knowledge with situated robot vision and perception, to allow robots to continuously extend their object knowledge beyond perceptual models. First, we presented an integrated system to suggest concept labels for unknown objects observed by a mobile robot. Our system stores the spatial contexts in which objects are observed and uses these to query a Web-based suggestion system to receive a list of possible concepts that could apply to the unknown object. These suggestions are based on the relatedness of the objects observed with the unknown object, and can be improved by filtering the results based on both frequency and spatial proximity. We evaluated our system data from real office observations and demonstrated how various filter parameters changed the match of the results to ground truth data.

Moreover, we have presented a framework for extracting manipulation-relevant knowledge about objects in the form of (binary) relations. The framework relies on a ranking measure that, given an object, ranks all entities that potentially stand in the relation in question to the given object. We rely on a representational approach that exploits distributional spaces to embed entities into low-dimensional spaces in which the ranking measure can be evaluated. We have presented results on two relations: the relation between an object and its prototypical

location (`locatedAt`) as well as the relation between an object and one of its intended uses (`usedFor`). As main contribution, we have presented a supervised approach based on a neural network that, instead of using the cosine similarity as measure of semantic relatedness, uses positive and negative examples to train a scoring function in a supervised fashion. As an avenue for future work, the generalizability of the proposed methods to a wider set of relations can be considered. In the context of manipulation-relevant knowledge for a robotic system, other interesting properties of an object include its prototypical size, weight, texture, and fragility.

In the second part of this chapter, we have addressed the task of lyrics processing. First, we addressed the segmentation on synchronized text-audio representations of songs. For the songs in the corpus DALI where the lyrics are aligned to the audio, we have derived a measure of alignment quality specific to our task of lyrics segmentation. Then, we have shown that exploiting both textual and audio-based features lead the employed Convolutional Neural Network-based model to significantly outperform the state-of-the-art system for lyrics segmentation that relies on purely text-based features. Moreover, we have shown that the advantage of a bimodal segment representation pertains even in the case where the alignment is noisy. This indicates that a lyrics segmentation model can be improved in most situations by enriching the segment representation by another modality (such as audio).

Second, we have defined and addressed the task of lyrics summarization. We have applied both generic unsupervised text summarization methods (TextRank and a topic-based method we called TopSum), and a method inspired by audio thumbnailing on 50k lyrics from the WASABI corpus. We have carried out an automatic evaluation on the produced summaries computing standard metrics in text summarization, and a human evaluation with 26 participants, showing that using a fitness measure transferred from the musicology literature, we can amend generic text summarization algorithms and produce better summaries. In future work, we plan to address the challenging task of abstractive summarization over song lyrics, with the goal of creating a summary of song texts in prose-style - more similar to what humans would do, using their own words.

Finally, we have described the WASABI dataset of songs, focusing in particular on the lyrics annotations resulting from the applications of the methods we proposed to extract relevant information from the lyrics. So far, lyrics annotations concern their structure segmentation, their topic, the explicitness of the lyrics content, the summary of a song and the emotions conveyed. Some of those annotation layers are provided for all the 1.73M songs included in the WASABI corpus, while some others apply to subsets of the corpus, due to various constraints previously described. In [20] the authors have studied how song writers influence each other. We aim to learn a model that detects the border between heavy influence and plagiarism.

In the last part of this chapter, we focused on events extraction from Twitter messages, and we described a model for detecting open-domain events from tweets by modeling relationships between NE mentions and terms in a directed graph. The proposed approach is unsupervised and can automatically detect fine-grained events without prior knowledge of the number or type of events. Our experiments on two gold-standard datasets show that the approach yields state-of-the-art results. In the future, we plan to investigate whether linking terms to ontologies (e.g., DBpedia, YAGO) can help in detecting different mentions of the same entity, as preliminarily shown in [158]. This can be used to reduce the density of the event graph. Another possible improvement would be to enrich the content of the tweets with information from external web pages resolving the URLs in the tweets.

As a second study on events extraction, we have described a framework to generate timelines of salient sub-events in sports games exploiting information contained in tweets. Experiments on a set of tweets collected during EURO 2016 proved that our approach accurately detects sub-events in sports games when compared to news on the same events reported by sports media. While previous approaches focused only on detecting the type of the most important

sub-events, we extract and model a richer set of information, including almost every type of sub-event and participants involved in the actions. Possible improvements for future work include the extension of our approach to other sports (e.g., American football).

Chapter 3

Natural language interaction with the Web of Data

This Chapter is dedicated to my contributions addressing the challenge of enhancing users' interactions with the web of data by mapping natural language expressions (e.g. user queries) with concepts and relations in a structured knowledge base. These research contributions fit the areas of Natural Language Processing and Semantic Web, and have been published in several venues:

- Elena Cabrio, Serena Villata, Alessio Palmero Aprosio (2017). A RADAR for information reconciliation in Question Answering systems over Linked Data. Semantic Web 8(4): 601-617 [93].
- Elena Cabrio, Serena Villata, Julien Cojan, Fabien Gandon (2014). Classifying Inconsistencies in DBpedia Multilingual Chapters, in Proceedings of the Language Resources and Evaluation Conference (LREC-2014), pp. 1443-1450 [95].
- Elena Cabrio, Julien Cojan, Fabien Gandon (2014). Mind the cultural gap: bridging language specific DBpedia chapters for Question Answering, in Philipp Cimiano, Paul Buitelaar (eds.) Towards the Multilingual Semantic Web, pp. 137-154, Springer Verlag [84].
- Elena Cabrio, Vivek Sachidananda, Raphael Troncy (2014) Boosting QAKiS with multimedia answer visualization, Proceedings of the Extended Semantic Web Conference (ESWC 2014) - Demo/poster paper [88].
- Elena Cabrio, Alessio Palmero Aprosio, Serena Villata (2014). Reconciling Information in DBpedia through a Question Answering System, Proceedings of the 13th International Semantic Web Conference (ISWC 2014). Demo paper [87].
- Elena Cabrio, Julien Cojan, Fabien Gandon, and Amine Hallili (2013). Querying multilingual DBpedia with QAKiS, in The Semantic Web: ESWC 2013 Satellite Events, Lecture Notes in Computer Science, Volume 7955, pp. 194-198 [85].
- Julien Cojan, Elena Cabrio, Fabien Gandon (2013). Filling the Gaps Among DBpedia Multilingual Chapters for Question Answering. Proceedings of the 5th Annual ACM Web Science Conference, pp. 33-42 [119].
- Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, Fabien Gandon (2012). Natural Language Interaction with the Web of Data by Mining its Textual Side, Intelligenza Artificiale. 6 (2012), pp. 121-133. Special Issue on Natural Language Processing in the Web Era. IOS Press [82].

- Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, Bernardo Magnini, Alberto Lavelli, Fabien Gandon (2012). QAKiS: an Open Domain QA System based on Relational Patterns, Proceedings of the 11th International Semantic Web Conference (ISWC 2012). Demo paper [83].

To enhance users' interactions with the web of data, query interfaces providing a flexible mapping between natural language expressions, and concepts and relations in structured knowledge bases (KB) are particularly relevant. In this direction, I have mainly focused on the automatic extraction of structured data from unstructured documents to populate RDF triple stores, and in a Question Answering (QA) setting, on the mapping of natural language expressions (e.g. user queries) with concepts and relations in a structured KB. More specifically, I have proposed and implemented i) the WikiFramework, i.e. a methodology to collect relational patterns in several languages, and ii) QAKiS, a system for open domain QA over linked data [83]. QAKiS (Question Answering wiKiframework-based System), allows end users to submit a query to an RDF triple store in English and obtain the answer in the same language, hiding the complexity of the non-intuitive formal query languages involved in the resolution process, but exploiting the expressiveness of these standards to scale to the huge amounts of available semantic data. QAKiS addresses the task of QA over structured KBs (e.g. language specific DBpedia chapters) where the relevant information is expressed also in unstructured form (e.g. Wikipedia pages). A crucial issue is the interpretation of the question to convert it into a corresponding formal query. Differently from the approaches developed at the time of this work, QAKiS implements a relation-based match, where fragments of the question are matched to relational textual patterns automatically collected from Wikipedia (i.e. the WikiFramework repository). Such relation-based matching provides more precision with respect to matching on single tokens.

To evaluate QAKiS on a comparable setting, we took part into QALD-2 and 3 evaluation campaigns in 2012 and 2013. The challenge (that was run until 2018) is aimed at any kind of QA system that mediates between a user, expressing his or her information need in natural language, and semantic data. A training and a test set of 100 natural language questions each are provided by the organizers. The version of QAKiS that took part into such challenges targets only questions containing a Named Entity related to the answer through one property of the ontology. However, QAKiS performances are in line with the results obtained by the other participating systems.

As introduced before, QAKiS is based on Wikipedia for patterns extraction. English, French and German DBpedia chapters are the RDF data sets to be queried using a natural language interface [85]. While querying at the same time different heterogeneous interlinked datasets, the system may receive different results for the same query. The returned results can be related by a wide range of heterogenous relations, e.g., one can be the specification of the other, an acronym of the other, etc. In other cases, such results can contain an inconsistent set of information about the same topic. A well-known example of such heterogenous interlinked datasets are language-specific DBpedia chapters, where the same information may be reported in different languages. Given the growing importance of multilingualism in the Semantic Web community, and in Question Answering over Linked Data in particular, we choose to apply information reconciliation to this scenario. We have therefore addressed the issue of reconciling information obtained by querying the SPARQL endpoints of language-specific DBpedia chapters integrating the proposed framework into QAKiS.

Moreover, in [88] we extended QAKiS to exploit the structured data and metadata describing multimedia content on the linked data to provide a richer and more complete answer to the user, combining textual information with other media content. A first step in this direction consisted in determining the best sources and media (image, audio, video, or a hybrid) to answer a query. Then, we have extended QAKiS output to include *i)* pictures from Wikipedia

Infoboxes, for instance to visualize images of people or places, *ii)* OpenStreetMap, to visualize maps and *iii)* YouTube, to visualize videos related to the answer.

This chapter is organized as follows: Section 3.1 describes the Question Answering system QAKiS, then Section 3.2 explains the method proposed to reconcile inconsistent information that can be collected from multilingual data sources. Section 3.3 describes QAKiS multimedia visualization, while Section 4.6 summarizes related work. Conclusions end the chapter.

3.1 QAKiS

To enhance users' interactions with the web of data, query interfaces providing a flexible mapping between natural language expressions, and concepts and relations in structured knowledge bases are becoming particularly relevant. In this section, we present QAKiS (Question Answering wiKiframework-based System), that allows end users to submit a query to an RDF triple store in English and obtain the answer in the same language, hiding the complexity of the non-intuitive formal query languages involved in the resolution process. At the same time, the expressiveness of these standards is exploited to scale to the huge amounts of available semantic data. In its current implementation, QAKiS addresses the task of QA over structured Knowledge Bases (KBs) (e.g. DBpedia) where the relevant information is expressed also in unstructured form (e.g. Wikipedia pages). Its major novelty with respect to the systems available at the time of this work, is to implement a relation-based match for question interpretation, to convert the user question into a query language (e.g. SPARQL). Most of the approaches up to then (for an overview, see [290]) based this conversion on some form of flexible matching between words of the question and concepts and relations of a triple store, disregarding the relevant context around a word, without which the match might be wrong. QAKiS tries instead first to establish a matching between fragments of the question and relational textual patterns automatically collected from Wikipedia. The underlying intuition is that a relation-based matching would provide more precision with respect to matching on single tokens, as done by competitors systems.

QAKiS demo is based on Wikipedia for patterns extraction. DBpedia is the RDF data set to be queried using a natural language interface.

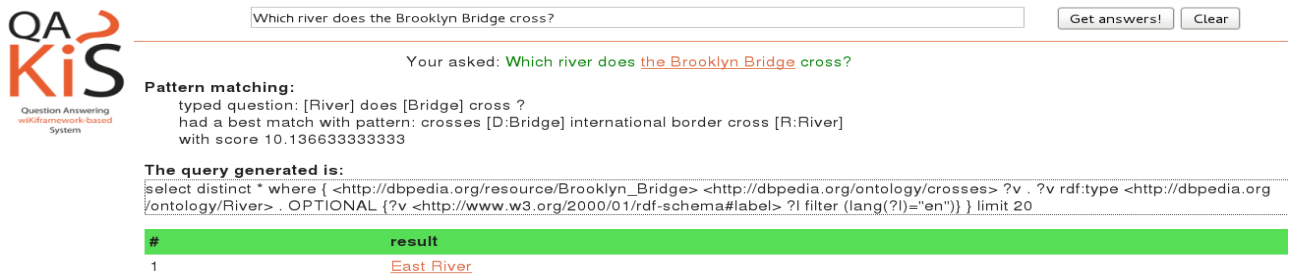


Figure 3.1: QAKiS demo interface (v1). The user can either write a question (or select among a list of examples) and click on *Get Answers!*. QAKiS outputs: *i)* the user question (the recognized Named Entity (NE) is linked to its DBpedia page), *ii)* the generated typed question (see Section 3.1.1), *iii)* the pattern matched, *iv)* the SPARQL query sent to the DBpedia SPARQL endpoint, and *v)* the answer (below the green rectangle *results*).

QAKiS makes use of relational patterns (automatically extracted from Wikipedia and collected in the WikiFramework repository [297]), that capture different ways to express a certain relation in a given language. For instance, the relation `crosses(Bridge,River)` can be expressed in English, among the others, by the following relational patterns: [`Bridge` crosses the `River`] and [`Bridge` spans over the `River`]. Assuming that there is a high probability that the

information in the Infobox is also expressed in the same Wikipedia page, the WikiFramework establishes a 4-step methodology to collect relational patterns in several languages for the DBpedia ontology relations (similarly to [192],[478]): *i)* a DBpedia relation is mapped with all the Wikipedia pages in which such relation is reported in the Infobox; *ii)* in such pages we collect all the sentences containing both the domain and the range of the relation; *iii)* all sentences for a given relation are extracted and the domain and range are replaced by the corresponding DBpedia ontology classes; *iv)* the patterns for each relation are clustered according to the lemmas between the domain and the range, and sorted according to their frequency.

3.1.1 System architecture

QAKiS is composed of four main modules (Fig. 3.2): *i)* the **query generator** takes the user question as input, generates the typed questions, and then generates the SPARQL queries from the retrieved patterns; *ii)* the **pattern matcher** takes as input a typed question, and retrieves the patterns (among those in the repository) matching it with the highest similarity; *iii)* the **sparql package** handles the queries to DBpedia; and *iv)* a **Named Entity (NE) Recognizer**. The multimedia answer generator module was added later, and will be described in Section 3.3.

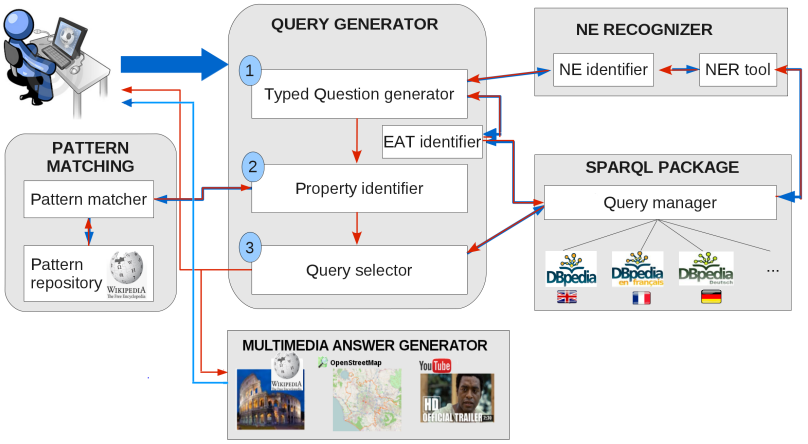
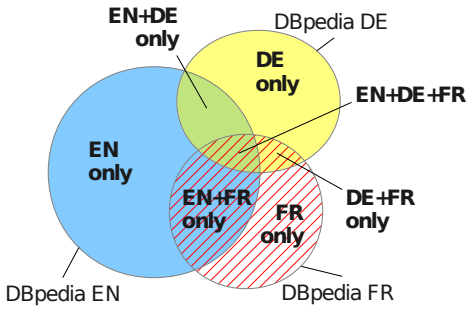


Figure 3.2: QAKiS workflow

The current version of QAKiS targets questions containing a NE related to the answer through one property of the ontology, as *Which river does the Brooklyn Bridge cross?*. Each question matches a single pattern (i.e. one relation).

Before running the *pattern matcher* component, the question target is identified combining the output of Stanford NE Recognizer, with a set of strategies that compare it with the instances labels in the DBpedia ontology. Then a *typed question* is generated by replacing the question keywords (e.g. who, where) and the NE by the types and supertypes. A Word Overlap algorithm is then applied to match such typed questions with the patterns for each relation. A similarity score is provided for each match: the highest represents the most likely relation. A set of patterns is retrieved by the pattern matcher component for each typed question, and sorted by decreasing matching score. For each of them, a set of SPARQL queries is generated and then sent to an endpoint for answer retrieval.

Multilingual DBpedia alignment. Multilingual DBpedia chapters[1] have been created following Wikipedia structure: each chapter contains therefore data extracted from Wikipedia in the corresponding language, and so reflects local specificity. Data from different DBpedia chapters are connected by several alignments: *i) instances* are aligned according to the inter-language links, that are created by Wikipedia editors to relate articles about the same topic in different languages; *ii) properties* that mostly come from template attributes, i.e. structured elements that can be included in Wikipedia pages so as to display structured information. These properties are aligned through mappings manually edited by the DBpedia community. Since QAKiS allows to query English, French and German DBpedia chapters, Figure 3.3 shows the additional information coverage provided by the mentioned DBpedia chapters. Areas **FR only**, **DE only** and **DE+FR only** correspond to aligned data made available by French and German DBpedia chapters.



Pairs (subject, property) with values in DBpedia chapters	Number
EN+FR+DE	110 685
EN+DE only	48 804
EN+FR only	408 125
DE+FR only	231 585
EN only	1 790 972
DE only	1 036 953
FR only	411 782

Figure 3.3: Relative coverage of English, German and French DBpedia chapters (in 2013).

QAKiS extension to query DBpedia multilingual chapters. QAKiS extension to query the ontology properties of multilingual DBpedia is integrated at the *SPARQL package* level. Instead of sending the query to English DBpedia only, the *Query manager* reformulates the queries and sends them to multiple DBpedia chapters. As only the English chapter contains labels in English, this change has no impact on the NE Recognition. The main difference is in the query selection step. As in the monolingual setting, patterns are taken iteratively by decreasing matching score, the generated query is then evaluated and if no results are found the next pattern is considered, and so on. However, as queries are evaluated on several DBpedia chapters, it is more likely to get results, terminating query selection with a higher matching score. Currently, the results of a SPARQL query are aggregated by the set union. Other strategies could be considered, as using a voting mechanism to select the most frequent answer, or enforcing a priority according to data provenance (e.g. English chapter could be considered as more reliable for questions related to English culture).

3.1.2 Experimental evaluation

QALD-2 dataset. Table 1 reports QAKiS's results on the QALD-2 data sets[2] (DBpedia track). For the demo, we focused on code optimization reducing QAKiS average processing time per question from 15 to 2 sec., w.r.t. the version used for the challenge.

Most of QAKiS' mistakes concern wrong relation assignment (i.e. wrong pattern matching). Another issue concerns questions ambiguity, i.e. the same surface forms can in fact refer to different relations in the DBpedia ontology. We plan to cluster relations with several patterns in common, to allow QAKiS to search among all the relations in the cluster.

[1] `http://wiki.dbpedia.org/Internationalization/Chapters`
[2] `http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/`

	Precision	*Recall*	*F-measure*	*# answered*	*# right answ.*	*# partially right*
train	0.476	0.479	0.477	40/100	17/40	4/40
test	0.39	0.37	0.38	35/100	11/35	4/35

Table 3.1: QAKiS performances on DBpedia data sets (participation to QALD-2)

The partially correct answers concern questions involving more than one relation: the actual version of the algorithm detects indeed only one of them, therefore we would need to improve the system to address questions with multiple (often nested) relations.

QALD-2 dataset reduced. Since QAKiS currently targets only questions containing a NE related to the answer through one property of the ontology (e.g. *In which military conflicts did Lawrence of Arabia participate?*), we extracted from QALD-2 test set[10] the set of questions corresponding to such criterion (i.e. 32 questions). The discarded questions require either some forms of reasoning (e.g. counting or ordering) on data, aggregation (from datasets different from DBpedia), involve n-relations, or they are boolean questions. We run both QAKiS$_{EN}$ (i.e. the system taking part into the challenge) and QAKiS$_{EN+FR}$ and QAKiS$_{EN+DE}$ (the versions enriched with the French and German DBpedia, respectively) on the reduced set of questions. Since the answer to QALD-2 questions can be retrieved in English DBpedia, we do not expect multilingual QAKiS to improve its performances. On the contrary, we want to verify that QAKiS performances do not decrease (due to the choice of the wrong relation triggered by a different pattern that finds an answer in multilingual DBpedia). Even if often extended QAKiS selects different patterns with respect to the original system, the selected relation is the same (except than in one case), meaning that generally performances are not worsen by the addition of multilingual DBpedia chapters.

Multilingual DBpedia. Since that at the time of this work no reference list of questions whose answers can be found in French or German DBpedia only existed, we created our list to evaluate the improvement in QAKiS's coverage as follows: *i)* we take the sample of 32 QALD-2 questions described before; *ii)* we extract the list of triples present in French and German DBpedia only; in each question we substitute the NE with another entity for which the asked relation can be found respectively in the French or German chapters only. For instance, for the QALD-2 question *How tall is Michael Jordan?*, we substitute the NE *Michael Jordan* with the entity *Margaret Simpson*, for which we know that the relation `height` is missing in English DBpedia, but is present in the French chapter. As a result, we obtain the question *How tall is Margaret Simpson?*, that we submit to QAKiS$_{EN+FR}$. Following the same procedure for German, in *Who developed Skype?* we substituted the NE *Skype* with the entity *IronPython*, obtaining the question *Who developed IronPython?*. The same procedure is applied for German.[3] For some properties (e.g. `Governor`, `Battle`), no additional links are provided by the multilingual chapters, so we discarded the questions asking for those relations. QAKiS precision on the new set of questions over French and German DBpedia is in line with QAKiS$_{EN}$ on English DBpedia ($\sim$ 50%). This evaluation did not have the goal to show improved performances of the extended version of QAKiS with respect to its precision, but to show that the integration of multilingual DBpedia chapters in the system is easily achievable, and that the expected improvements on its coverage are really promising and worth exploring (see Figure 3.3). To double-check, we run the same set of questions on QAKiS$_{EN}$, and in no cases it was able to detect the correct answer, as expected.

[3]The obtained set of transformed questions is available online at `http://dbpedia.inria.fr/qakis/`.

3.2 RADAR 2.0: a framework for information reconciliation

In the Web of Data, it is possible to retrieve heterogeneous information items concerning a single real-world object coming from different data sources, e.g., the results of a single SPARQL query on different endpoints. It is not always the case that these results are identical, it may happen that they conflict with each other, or they may be linked by some other relation like a specification. The automated detection of the kind of relationship holding between different instances of a single object with the goal of reconciling them is an open problem for consuming information in the Web of Data. In particular, this problem arises while querying the language-specific chapters of DBpedia [310]. Such chapters, well connected through Wikipedia instance interlinking, can in fact contain different information with respect to the English version. Assuming we wish to query a set of language-specific DBpedia SPARQL endpoints with the same query, the answers we collect can be either identical, or in some kind of specification relation, or they can be contradictory. Consider for instance the following example: we query a set of language-specific DBpedia chapters about *How tall is the soccer player Stefano Tacconi?*, receiving the following information: 1.88 from the Italian chapter and the German one, 1.93 from the French chapter, and 1.90 from the English one. How can I know what is the "correct" (or better, the more reliable) information, knowing that the height of a person is unique? Addressing such kind of issues is the goal of the present section. More precisely, in this section, we answer the research question:

- How to reconcile information provided by the language-specific chapters of DBpedia?

This open issue is particularly relevant to Question Answering (QA) systems over DBpedia [290], where the user expects a unique (ideally correct) answer to her factual natural language question. A QA system querying different data sources needs to weight them in an appropriate way to evaluate the information items they provide accordingly. In this scenario, another open problem is how to explain and justify the answer the system provides to the user in such a way that the overall QA system appears transparent and, as a consequence, more reliable. Thus, our research question breaks down into the following subquestions: *i)* How to automatically detect the relationships holding between information items returned by different language-specific chapters of DBpedia? *ii)* How to compute the reliability degree of such information items to provide a unique answer? *iii)* How to justify and explain the answer the QA system returns to the user?

First, we need to classify the relations connecting each piece of information to the others returned by the different data sources, i.e., the SPARQL endpoints of the language-specific DBpedia chapters. To this purpose, we propose a categorization of the relations existing between different information items retrieved with a unique SPARQL query [95] (described later on in this section). At the time of this work, such categorization was the only one considering linguistic, fine-grained relations among the information items returned by language-specific DBpedia chapters, given a certain query. This categorization considers ten *positive* relations among heterogeneous information items (referring to widely accepted linguistic categories in the literature), and three *negative* relations meaning inconsistency. Starting from this categorization, we propose the RADAR (ReconciliAtion of Dbpedia through ARgumentation) framework, that adopts a classification method to return the relation holding between two information items. This first step results in a graph-based representation of the result set where each information item is a node, and edges represent the identified relations.

Second, we adopt *argumentation theory* [151], a suitable technique for reasoning about conflicting information, to assess the acceptability degree of the information items, depending on the relation holding between them and the trustworthiness of their information source [131].

Roughly, an abstract argumentation framework is a directed labeled graph whose nodes are the arguments and the edges represent a *conflict* relation. Since positive relations among the arguments may hold as well, we rely on bipolar argumentation [102] that considers also a *positive* support relation.

Third, the graph of the result set obtained after the classification step, together with the acceptability degree of each information item obtained after the argumentation step, is used to justify and explain the resulting information ranking (i.e., the order in which the answers are returned to the user).

We evaluate our approach as standalone (i.e., over a set of heterogeneous values extracted from a set of language-specific DBpedia chapters), and through its integration in the QA system QAKiS, described in Section 3.1. The reconciliation module is embedded to provide a (possibly unique) answer whose acceptability degree is over a given threshold, and the graph structure linking the different answers highlights the underlying justification. Moreover, RADAR is applied to over 300 DBpedia properties in 15 languages, and the obtained resource of reconciled DBpedia language-specific chapters is released.

Even if information reconciliation is a way to enhance Linked Data quality, we do not address the issue of Linked Data quality assessment and fusion [311, 73], nor ontology alignment. Finally, argumentation theory in this work is not exploited to find agreements over ontology alignments [148]. Note that our approach is intended to reconcile and explain the answer of the system to the user. Ontology alignment cannot be exploited to generate such a kind of explanations. This is why we decided to rely on argumentation theory that is a way to exchange and explain viewpoints. In this work, we have addressed the open problem of reconciling and explaining a result set from language-specific DBpedia chapters using well-known conflict detection and explanation techniques, i.e., argumentation theory.

3.2.1 Framework description

The RADAR 2.0 (ReconciliAtion of Dbpedia through ARgumentation) framework for information reconciliation is composed by three main modules (see Figure 3.4). It takes as input a collection of results from one SPARQL query raised against the SPARQL endpoints of the language-specific DBpedia chapters. Given such result set, RADAR retrieves two kinds of information: *(i)* the sources proposing each particular element of the result set, and *(ii)* the elements of the result set themselves. The first module of RADAR (module A, Figure 3.4) takes each information source, and following two different heuristics, assigns a *confidence degree* to the source. Such confidence degree will affect the reconciliation in particular with respect to the possible inconsistencies: information proposed by the more reliable source will obtain a higher acceptability degree. The second module of RADAR (module B, Figure 3.4) instead starts from the result set, and it matches every element with all the other returned elements, detecting the kind of relation holding between these two elements. The result of such module is a graph composed by the elements of the result set connected with each other by the relations of our categorization. Both the sources associated with a confidence score and the result set in the form of a graph are then provided to the third module of RADAR, the argumentation one (module C, Figure 3.4). The aim of such module is to reconcile the result set. The module considers all positive relations as a *support* relation and all negative relations as an *attack* relation, building a bipolar argumentation graph where each element of the result set is seen as an argument. Finally, adopting a bipolar fuzzy labeling algorithm relying on the confidence of the sources to decide the acceptability of the information, the module returns the acceptability degree of each argument, i.e., element of the result set. The output of the RADAR framework is twofold. First, it returns the acceptable elements (a threshold is adopted), and second the graph of the result set is provided, where each element is connected to the others

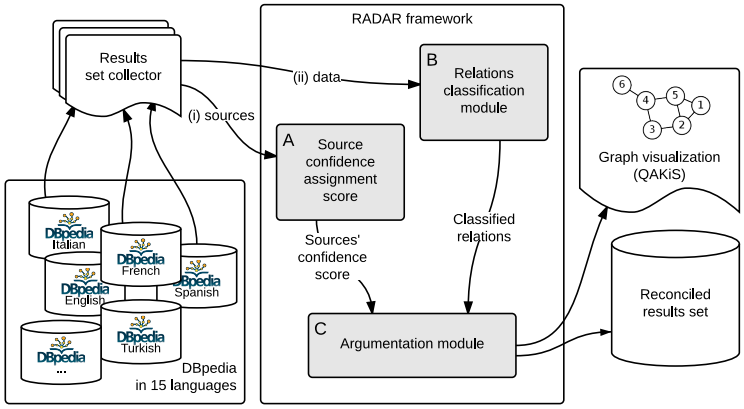


Figure 3.4: RADAR 2.0 framework architecture.

by the identified relations (i.e., the explanation about the choice of the acceptable arguments returned).

Assigning a confidence score to the source

Language-specific DBpedia chapters can contain different information on particular topics, e.g. providing more or more specific information. Moreover, the knowledge of certain instances and the conceptualization of certain relations can be culturally biased. For instance, we expect to have more precise (and possibly more reliable) information on the Italian actor Antonio Albanese on the Italian DBpedia, than on the English or on the French ones.

To trust and reward the data sources, we need to calculate the reliability of the source with respect to the contained information items. In [86], an a priori confidence score is assigned to the endpoints according to their dimensions and solidity in terms of maintenance (the English chapter is assumed to be more reliable than the others on all values, but this is not always the case). RADAR 2.0 assigns, instead, a confidence score to the DBpedia language-specific chapter depending on the queried entity, according to the following two criteria:

- *Wikipedia page length.* The chapter of the longest language-specific Wikipedia page describing the queried entity is considered as fully trustworthy (i.e., it is assigned with a score of 1) while the others are considered less trustworthy (i.e., they are associated with a score < 1). In choosing such heuristic, we followed [60] that demonstrates that the article length is a very good predictor of its precision. The length is calculated on the Wikipedia dump of the considered language (# of characters in the text, ignoring image tags and tables). Thus, the longest page is assigned a score equal to 1, and a proportional score is assigned to the other chapters.

- *Entity geo-localization.* The chapter of the language spoken in the places linked to the page of the entity is considered as fully trustworthy (i.e., it is assigned with a score of 1) while the others are considered less trustworthy (i.e., they are associated with a score < 1). We assume that if an entity belongs to a certain place or is frequently referred to it, it is more likely that the DBpedia chapter of such country contains updated and reliable information. All Wikipedia page hyperlinks are considered, and their presence in GeoNames[4] is checked. If existing, the prevalent language in the place (following the GeoNames matching country-language[5]) is extracted, and to the corresponding chapter a

[4] `http://www.geonames.org/`

[5] Such table connecting a country with its language can be found here: `http://download.geonames.org/export/dump/countryInfo.txt`.

score equal to 1 is assigned. As for page length, a proportional score is then assigned to the other chapters (i.e. if an entity has e.g. 10 links to places in Italy and 2 to places in Germany, the score assigned to the Italian DBpedia chapter is 1, while for the German chapter is 0.2).

Such metrics (the appropriateness of which for our purposes has been tested on the development set) are then summed and normalized with a score ranging from 0 to 1, where 0 is the least reliable chapter for a certain entity and 1 is the most reliable one. The obtained scores are then considered by the argumentation module for information reconciliation.

Relations classification

We propose a classification of the semantic relations holding among the different instances obtained by querying a set of language-specific DBpedia chapters with a certain query. More precisely, such categories correspond to the lexical and discourse relations holding among heterogeneous instances obtained querying two DBpedia chapters at a time, given a subject and an ontological property. In the following, we list the positive relations between values resulting from the data-driven study we carried out and described in [95]. Then, in parallel, we describe how RADAR 2.0 addresses the automatic classification of such relations.

Identity i.e., same value but in different languages (missing `owl:sameAs` link in DBpedia). E.g., `Dairy product` vs `Produits laitiers`

Acronym i.e., initial components in a phrase or a word. E.g., `PSDB` vs `Partito della Social Democrazia Brasiliana`

Disambiguated entity i.e., a value contains in the name the class of the entity. E.g., `Michael Lewis (Author)` vs `Michael Lewis`

Coreference i.e., an expression referring to another expression describing the same thing (in particular, non normalized expressions). E.g., `William Burroughs` vs `William S. Burroughs`

Given the high similarity among the relations belonging to these categories, we cluster them into a unique category called *surface variants* of the same entity. Given two entities, RADAR automatically detects the *surface variants* relation among them, if one of the following strategies is applicable: cross-lingual links[6], text identity (i.e. string matching), Wiki redirection and disambiguation pages.

Geo-specification i.e., ontological geographical knowledge. E.g., `Queensland` vs `Australia`

Renaming i.e., reformulation of the same entity name in time. E.g., `Edo`, old name of `Tokyo`

In [95], we defined *renaming* as referring only to geographical renaming. For this reason, we merge it to the category *geo-specification*. RADAR classifies a relation among two entities as falling inside this category when in the GeoNames one entity is contained in the other one (*geo-specification* is a directional relation between two entities). We also consider the alternative

[6]Based on WikiData, a free knowledge base that can be read and edited by humans and machines alike, `http://www.wikidata.org/`, where data entered in any language is immediately available in all other languages. In WikiData, each entity has the same ID in all languages for which a Wikipedia page exists, allowing us to overcome the problem of missing `owl:sameAs` links in DBpedia (that was an issue in DBpedia versions prior to 3.9). Moreover, WikiData is constantly updated (we use April 2014 release).

names gazette included in GeoNames, and geographical information extracted from a set of English Wikipedia infoboxes, such as `Infobox former country`[7] or `Infobox settlement`.

Meronymy i.e., a constituent part of, or a member of something. E.g., `Justicialist Party` is a part of `Front for Victory`

Hyponymy i.e., relation between a specific and a general word when the latter is implied by the former. E.g., `alluminio` vs `metal`

Metonymy i.e., a name of a thing/concept for that of the thing/concept meant. E.g., `Joseph Hanna` vs `Hanna-Barbera`

Identity:stage name i.e., pen/stage names pointing to the same entity. E.g., `Lemony Snicket` vs `Daniel Handler`

We cluster such semantic relations into a category called *inclusion*.[8] To detect this category of relations, RADAR exploits a set of features extracted from:

MusicBrainz[9] to detect when a musician plays in a band, and when a label is owned by a bigger label.

BNCF (Biblioteca Nazionale Centrale di Firenze) Thesaurus[10] for the broader term relation between common names.

DBpedia, in particular the datasets connecting Wikipedia, GeoNames and MusicBrainz trough the `owl:sameAs` relation.

WikiData for the *part of*, *subclass of* and *instance of* relations. It contains links to GeoNames, BNCF and MusicBrainz, integrating DBpedia `owl:sameAs`.

Wikipedia contains hierarchical information in: infoboxes (e.g. property `parent` for companies, `product` for goods, `alter ego` for biographies), categories (e.g., `Gibson guitars`), "see also" sections and links in the first sentence (e.g., *Skype was acquired by [United States]-based [Microsoft Corporation]*).

Inclusion is a directional relation between two entities (the rules we apply to detect *meronymy*, *hyponymy* and *stage name* allow us to track the direction of the relation, i.e. if $a \rightarrow b$, or $b \rightarrow a$).

Moreover, in the classification proposed in [95], the following negative relations (i.e., values mismatches) among possibly inconsistent data are identified:

Text mismatch i.e. unrelated entity. E.g., `Palermo` vs `Modene`

Date mismatch i.e. different date for the same event. E.g., `1215-04-25` vs `1214- 04-25`

[7] For instance, we extract the property "today" connecting historical entity names with the current ones (reconcilable with GeoNames). We used Wikipedia dumps.

[8] Royo [397] defines both relations of *meronymy* and *hyponymy* as relations of *inclusion*, although they differ in the kind of inclusion defined (hyponymy is a relation of the kind "B is a type of A", while meronymy relates a whole with its different parts or members). Slightly extending Royo's definition, we joined to this category also the relation of *metonymy*, a figure of speech scarcely detectable by automatic systems due to its complexity (and *stage name*, that can be considered as a particular case of *metonymy*, i.e., the name of the character for the person herself).

[9] `http://musicbrainz.org/`

[10] `http://thes.bncf.firenze.sbn.it/`

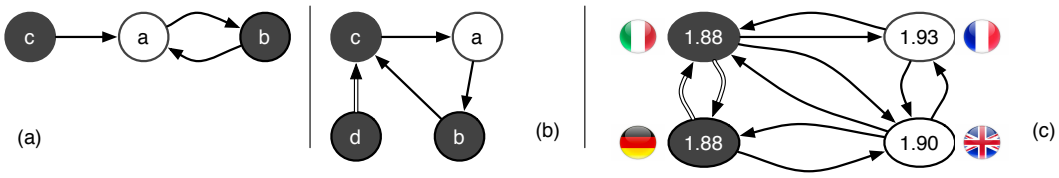


Figure 3.5: Example of (a) an *AF*, (b) a bipolar *AF*, and (c) example provided in the introduction modeled as a bipolar *AF*, where single lines represent attacks and double lines represent support.

Numerical mismatch i.e. different numerical values. E.g., `1.91` vs `1.8`

RADAR labels a relation between instances (i.e., URIs) as negative, if every attempt to find one of the positive relations described above fails (i.e., negation as a failure). For numerical values, a *numerical mismatch* identifies different values.[11]

The reader may argue that a machine learning approach could have been applied to this task, but a supervised approach would have required an annotated dataset to learn the features. Unfortunately, at the moment there is no such training set available to the research community. Moreover, given the fact that our goal is to produce a resource as precise as possible for future reuse, the implementation of a rule-based approach allows us to tune RADAR to reward precision in our experiments, in order to accomplish our purpose.

Argumentation-based information reconciliation

This subsection begins with a brief overview of abstract argumentation theory, and then we detail the RADAR 2.0 argumentation module.

An abstract argumentation framework (AF) [151] aims at representing conflicts among elements called *arguments*, whose role is determined only by their relation with other arguments. An AF encodes, through the conflict (i.e., *attack*) relation, the existing conflicts within a set of arguments. It is then interesting to identify the conflict outcomes, which, roughly speaking, means determining which arguments should be accepted, and which arguments should be rejected, according to some reasonable criterion.

The set of accepted arguments of an argumentation framework consists of a set of arguments that does not contain an argument conflicting with another argument in the set. Dung [151] presents several acceptability semantics that produce zero, one, or several *consistent* sets of accepted arguments. Roughly, an argument is *accepted* (i.e., labelled *in*) if all the arguments attacking it are rejected, and it is *rejected* (i.e., labelled *out*) if it has at least an argument attacking it which is accepted. Figure 3.5.a shows an example of an AF. The arguments are visualized as nodes of the argumentation graph, and the attack relation is visualized as edges. Gray arguments are the accepted ones. Using Dung's admissibility-based semantics [151], the set of accepted arguments is $\{b, c\}$. For more details about acceptability semantics, we refer the reader to Baroni at al. [32].

However, associating a *crisp* label, i.e., *in* or *out*, to the arguments is limiting in a number of real life situations where a numerical value expressing the acceptability degree of each argument is required [153, 131, 236]. In particular, da Costa Pereira et al. [131] have proposed a fuzzy labeling algorithm to account for the fact that arguments may originate from sources that are trusted only to a certain degree. They define a fuzzy labeling for argument A as $\alpha(A) = \min\{\mathcal{A}(A), 1 - \max_{B:B\rightarrow A} \alpha(B)\}$ where $\mathcal{A}(A)$ is given by the trust degree of the most reliable

[11] At the moment no tolerance is admitted, if e.g. the height of a person differs of few millimeters in two DBpedia chapters, the relation is labeled as *numerical mismatch*. We plan to add such tolerance for information reconciliation as future work.

source that offers argument A, and argument B is an argument attacking A. We say that $\alpha(A)$ is the fuzzy label of argument A. Consider the example in Figure 3.5.a, if we have $\mathcal{A}(a) = \mathcal{A}(b) = \mathcal{A}(c) = 0.8$, then the algorithm returns the following labeling: $\alpha(a) = 0.2$ and $\alpha(c) = \alpha(b) = 0.8$.

Since we want to take into account the confidence associated with the information sources to compute the acceptability degree of arguments, we rely on the computation of fuzzy confidence-based degrees of acceptability. As the fuzzy labeling algorithm [131] exploits a scenario where the arguments are connected by an attack relation only, in Cabrio et al. [86] we have proposed a bipolar version of this algorithm, to consider also a positive, i.e., support, relation among the arguments (bipolar AFs) for the computation of the fuzzy labels of the arguments.

Let $\mathcal{A}$ be a fuzzy set of trustful arguments, and $\mathcal{A}(A)$ be the membership degree of argument A in $\mathcal{A}$, we have that $\mathcal{A}(A)$ is given by the trust degree of the most reliable (i.e., trusted) source that offers argument A[12], and it is defined as follows: $\mathcal{A}(A) = \max_{s \in \text{src}(A)} \tau_s$ where τ_s is the degree to which source $s \in \text{src}(A)$ is evaluated as reliable. The starting confidence degree associated with the sources is provided by RADAR's first module. The bipolar fuzzy labeling algorithm [86] assumes that the following two constraints hold: *(i)* an argument cannot attack and support another argument at the same time, and *(ii)* an argument cannot support an argument attacking it, and vice versa. These constraints underlie the construction of the bipolar AF itself. In the following, the attack relation is represented with $\rightarrow$, and the support relation with $\Rightarrow$.

Definition 1 *Let $\langle \mathcal{A}, \rightarrow, \Rightarrow \rangle$ be an abstract bipolar argumentation framework where $\mathcal{A}$ is a fuzzy set of (trustful) arguments, $\rightarrow \subseteq \mathcal{A} \times \mathcal{A}$ and $\Rightarrow \subseteq \mathcal{A} \times \mathcal{A}$ are two binary relations called attack and support, respectively. A bipolar fuzzy labeling is a total function $\alpha : \mathcal{A} \rightarrow [0, 1]$.*

Such an α may also be regarded as (the membership function of) the fuzzy set of acceptable arguments where the label $\alpha(A) = 0$ means that the argument is outright unacceptable, and $\alpha(A) = 1$ means the argument is fully acceptable. All cases in between provide the degree of the acceptability of the arguments which may be considered accepted in the end, if they exceed a certain threshold.

A bipolar fuzzy labeling is defined as follows[13], where argument B is an argument attacking A and C is an argument supporting A:

Definition 2 *(Bipolar Fuzzy Labeling) A total function $\alpha : \mathcal{A} \rightarrow [0, 1]$ is a bipolar fuzzy labeling iff, for all arguments A, $\alpha(A) = avg\{\min\{\mathcal{A}(A), 1 - \max_{B:B \rightarrow A} \alpha(B)\}; \max_{C:C \Rightarrow A} \alpha(C)\}$.*

When the argumentation module receives the elements of the result set linked by the appropriate relation and the confidence degree associated to each source, the bipolar fuzzy labeling algorithm is applied to the argumentation framework to obtain the acceptability degree of each argument. In case of cyclic graphs, the algorithm starts with the assignment of the trustworthiness degree of the source to the node, and then the value converges in a finite number of steps to the final label. Note that when the argumentation framework is composed by a cycle only, then all labels become equal to 0.5.

Consider the example in Figure 3.5.b, if we have $\mathcal{A}(a) = \mathcal{A}(d) = 1$, $\mathcal{A}(b) = 0.4$ and $\mathcal{A}(c) = 0.2$, then the fuzzy labeling algorithm returns the following labels: $\alpha(a) = \alpha(b) = 0.4$, $\alpha(c) = 0.6$, and $\alpha(d) = 1$. The step by step computation of the labels is shown in Table 3.2. Figure 3.5.c shows how the example provided in the introduction is modeled as a bipolar argumentation framework, where we expect the Italian DBpedia chapter to be the most reliable

[12]We follow da Costa Pereira et al. [131] choosing the max operator ("optimistic" assignment of the labels), but the min operator may be preferred for a pessimistic assignment.

[13]For more details about the bipolar fuzzy labeling algorithm, see Cabrio et al. [86].

Table 3.2: BAF: $a \rightarrow b, b \rightarrow c, c \rightarrow a, d \Rightarrow c$

t	$\alpha_t(a)$	$\alpha_t(b)$	$\alpha_t(c)$	$\alpha_t(d)$
0	1	0.4	0.2	**1**
1	0.9	0.2	**0.6**	↓
2	0.65	0.15	↓	
3	0.52	0.25		
4	0.46	0.36		
5	0.43	**0.4**		
6	0.41	↓		
7	**0.4**			
8	↓			

one, given that Stefano Tacconi is an Italian soccer player. The result returned by the bipolar argumentation framework is that the trusted answer is 1.88. A more precise instantiation of this example in the QA system is shown in the next section.

The fact that an argumentation framework can be used to provide an explanation and justify positions is witnessed by a number of applications in different contexts [45], like for instance practical reasoning [464], legal reasoning [46, 52], medical diagnosis [237]. This is the reason why we choose this formalism to reconcile information, compute the set of reliable information items, and finally justify this result. Other possible solutions would be (weighted) voting mechanisms, where the preferences of some voters, i.e., the most reliable information sources, carry more weight than the preferences of other voters. However, voting mechanisms do not consider the presence of (positive and negative) relations among the items within the list, and no justification beyond the basic trustworthiness of the sources is provided to motivate the ranking of the information items.

Notice that argumentation is needed in our use case because we have to take into account the trustworthiness of the information sources, and it provides an explanation of the ranking, which is not possible with simple majority voting. Argumentation theory, used as a conflict detection technique, allows us to detect inconsistencies and consider the trustworthiness evaluation of the information sources, as well as proposing a single answer to the users. As far as we know, RADAR integrated in QAKiS is the first example of QA over Linked Data system coping with this problem and providing a solution. Simpler methods would not allow to cover both aspects mentioned above. We use bipolar argumentation instead of non-bipolar argumentation because we have not only the negative conflict relation but also the positive support relation among the elements of the result set.

3.2.2 Experimental setting and evaluation

In the following, we describe the dataset on which we evaluate the RADAR framework, and we discuss the obtained results. Moreover, we describe the resource of reconciled DBpedia information we create and release.

Dataset

To evaluate the RADAR framework, we created an annotated dataset of possibly inconsistent information in DBpedia language-specific chapters to our knowledge [95]. It is composed of 400 annotated pairs of values (extracted from English, French and Italian DBpedia chapters), a sample that is assumed to be representative of the linguistic relations holding between values in DBpedia chapters. Note that the size of the DBpedia chapter does not bias the type of relations

identified among the values, nor their distribution, meaning that given a specific property, each DBpedia chapter deals with that property in the same way. We randomly divided such dataset into a development (to tune RADAR) and a test set, keeping the proportion among the distribution of categories.[14] Table 3.3 reports on the dataset statistics, and shows how many annotated relations belong to each of the categories described before in this section.

Table 3.3: Statistics on the dataset used for RADAR 2.0 evaluation

Dataset	*# triples*	*# annotated positive relations*			*# annotated negative relations*		
		Surface-form	Geo-specific.	Inclusion	Text mismatch	Date mismatch	Numerical mismatch
Dev set	104	28	18	20	13	13	12
Test set	295	84	48	55	36	37	35
Total	399	112	66	75	49	50	47

3.2.3 Results and discussion

Table 3.4 shows the results obtained by RADAR on the relation classification task on the test set. As baseline, we apply an algorithm exploiting only cross-lingual links (using WikiData), and exact string matching. Since we want to produce a resource as precise as possible for future reuse, RADAR has been tuned to reward precision (i.e., so that it does not generate false positives for a category), at the expense of recall (errors follow from the generation of false negatives for positive classes). As expected, the highest recall is obtained on the *surface form* category (our baseline performs even better than RADAR on such category). The *geo-specification* category has the lowest recall, either due to missing alignments between DBpedia and GeoNames (e.g. Ixelles and Bruxelles are not connected in GeoNames), or to the values complexity in the *renaming* subcategory (e.g., Paris vs First French Empire, or Harburg (quarter) vs Hambourg). In general, the results obtained are quite satisfying, fostering future work in this direction.

Table 3.4: Results of the system on relation classification

System	*Relation category*	*Precision*	*Recall*	F_1
RADAR 2.0	*surface form*	0.91	0.83	0.87
	geo-specification	0.94	0.60	0.73
	inclusion	0.86	0.69	0.77
	overall positive	1.00	0.74	0.85
	text mismatch	0.45	1	0.62
baseline	*surface form*	1.00	0.44	0.61
	geo-specification	0.00	0.00	0.00
	inclusion	0.00	0.00	0.00
	overall positive	1.00	0.21	0.35
	text mismatch	0.21	1	0.35

Since we consider *text mismatch* as a negative class (Section 3.2.1), it includes the cases in which RADAR fails to correctly classify a pair of values into one of the positive classes. For date and numerical mismatches, $F_1 = 1$ (detecting them is actually a trivial task, and therefore they are not included in Table 3.4. See footnote 11). *Overall positive* means that RADAR correctly understands the fact that the different answers to a certain query are all correct and

[14]The dataset is available at `http://www.airpedia.org/radar-1.0.nt.bz2`. The original work is based on DBpedia 3.9, but we updated it to DBpedia 2014. Thus, we deleted one pair, since the DBpedia page of one of the annotated entities does not exist anymore.

not conflicting. RADAR precision in this case is 1, and it is important to underline this aspect in the evaluation, since this confirms the reliability of the released reconciled DBpedia in this respect. The overall positive result is higher than the partial results because in the precision of partial values we include the fact that if e.g., a *surface form* relation is wrongly labeled as *geo-specification*, we consider this mistake both as a false negative for *surface form*, and as a false positive for *geo-specification*. This means that RADAR is very precise in assigning positive relations, but it could provide a less precise classification into finer-grained categories.

Reconciled DBpedia resource

We applied RADAR 2.0 on 300 DBpedia properties - the most frequent in terms of chapters mapping such properties, corresponding to 47.8% of all properties in DBpedia. We considered ∼5M Wikipedia entities. The outcoming resource, a sort of *universal DBpedia*, counts ∼50M of reconciled triples from 15 DBpedia chapters: Bulgarian, Catalan, Czech, German, English, Spanish, French, Hungarian, Indonesian, Italian, Dutch, Polish, Portuguese, Slovenian, Turkish. Notice that we did not consider the endpoint availability as a requirement to choose the DBpedia chapters: data are directly extracted from the resource.

For functional properties, the RADAR framework is applied as previously described. In contrast, the strategy to reconcile the values of non-functional properties is slightly different: when a list of values is admitted (e.g. for properties `child` or `instruments`), RADAR merges the list of the elements provided by the DBpedia chapters, and ranks them with respect to the confidence assigned to their source, after reconciling positive relations only (there is no way for lists to understand if an element is incorrect or just missing, e.g. in the list of the instruments played by John Lennon). But since the distinction between functional/non-functional properties is not precise in DBpedia, we manually annotated the 300 properties with respect to this classification, to allow RADAR to apply the correct reconciliation strategy, and to produce a reliable resource. In total, we reconciled 3.2 million functional property values, with an average accuracy computed from the precision and recall reported in Table 3.4.

Moreover, we carried out a merge and a light-weight reconciliation of DBpedia classes applying the strategy called "DBpedia CL" in [14] where "CL" stands for cross-language (e.g., *Michael Jackson* is classified as a `Person` in the Italian and German DBpedia chapters, an `Artist` in the English DBpedia and a `MusicalArtist` in the Spanish DBpedia. As `Person`, `Artist` and `MusicalArtist` lie on the same path from the root of the DBpedia ontology, all of them are kept and used to classify *Michael Jackson*.

3.2.4 Integrating RADAR in a QA system

We integrate RADAR into a QA system over language-specific DBpedia chapters, given the importance that information reconciliation has in this context. Indeed, a user expects a unique (and preferably correct) answer to her factual natural language question, and would not trust a system providing her with different and possibly inconsistent answers coming out of a black box. A QA system querying different data sources needs therefore to weight in an appropriate way such sources in order to evaluate the information items they provide accordingly.

As QA system we selected QAKiS, described in Section 3.1. As explained before, in QAKiS the SPARQL query created after the question interpretation phase is sent to the SPARQL endpoints of the language-specific DBpedia chapters (i.e., English, French, German and Italian) for answer retrieval. The set of retrieved answers from each endpoint is then sent to RADAR 2.0 for answer reconciliation.

To test RADAR integration into QAKiS, the user can select the DBpedia chapter she wants to query besides English (that must be selected as it is needed for NE recognition), i.e., French, German or Italian DBpedia. Then the user can either write a question or select among a list of

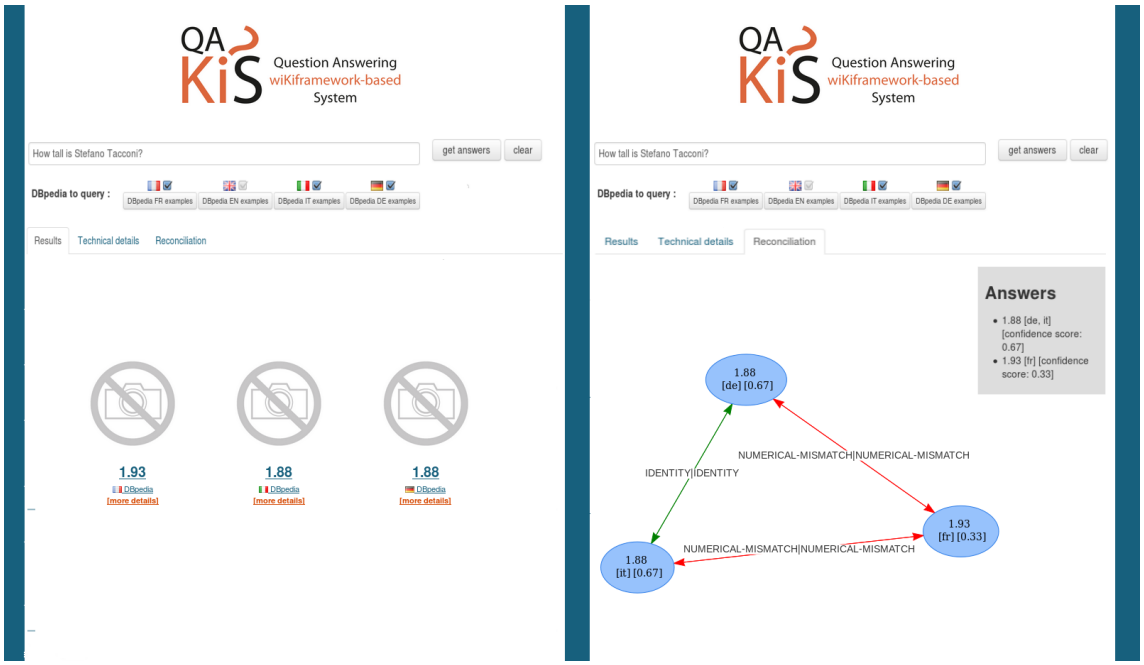


Figure 3.6: QAKiS + RADAR demo (functional properties)

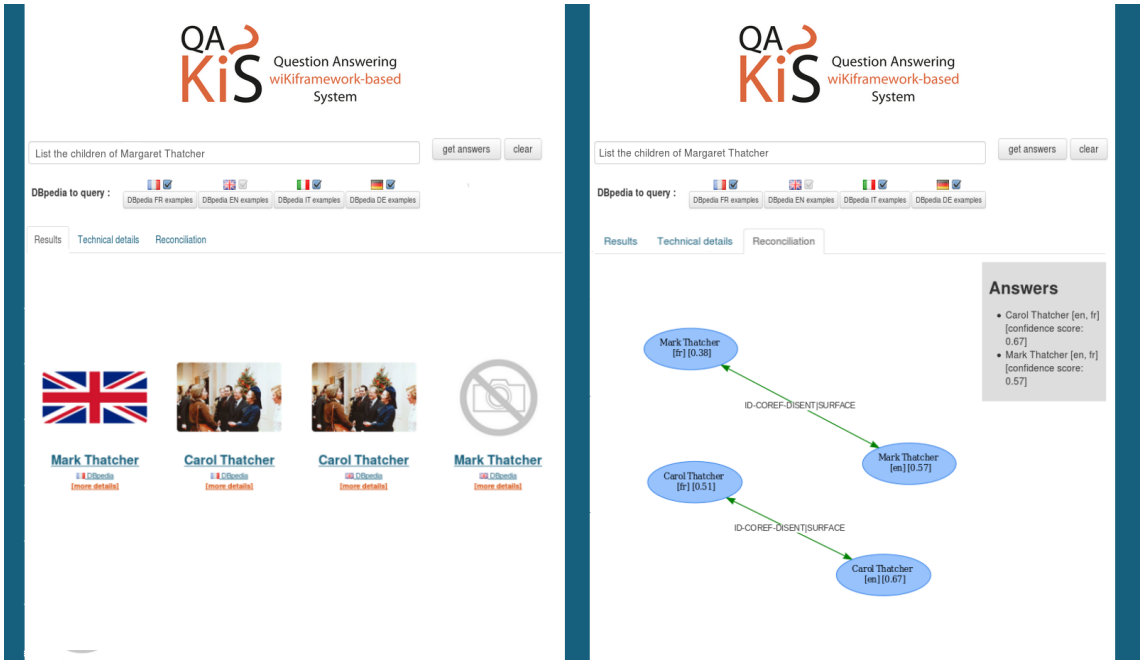


Figure 3.7: QAKiS + RADAR demo (non-functional properties)

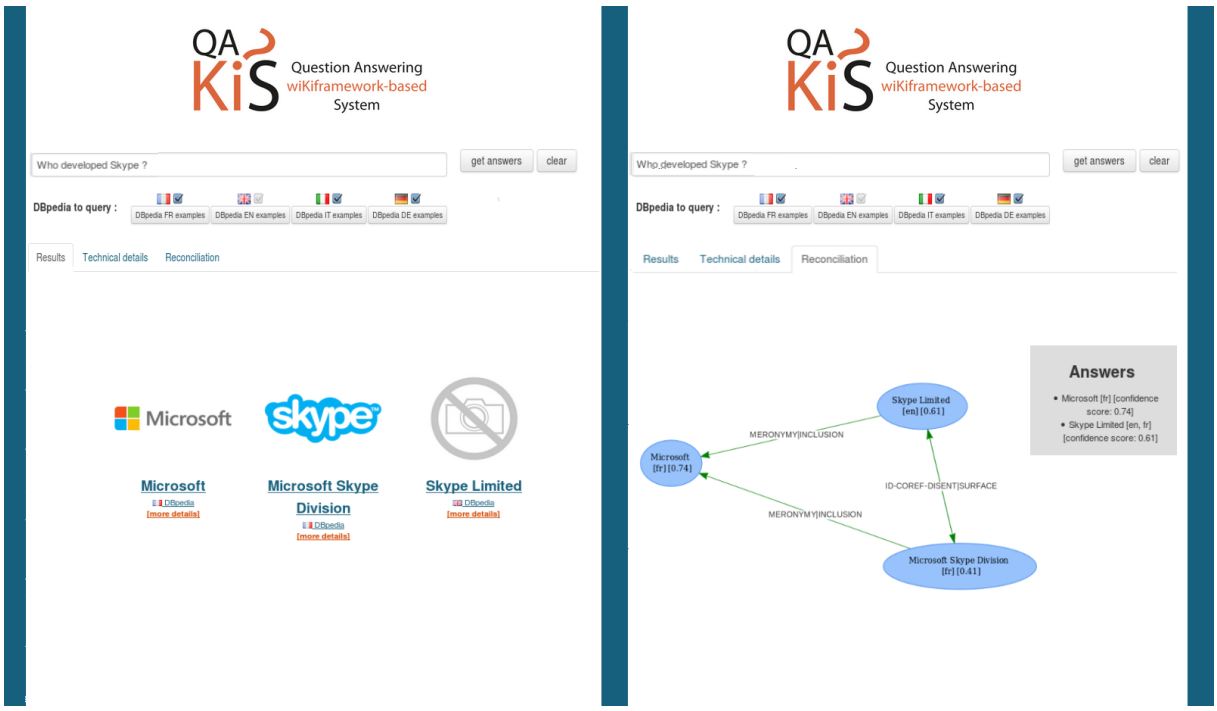


Figure 3.8: Example about the question *Who developed Skype?*

examples. Clicking on the tab *Reconciliation*, a graph with the answers provided by the different endpoints and the relations among them is shown to the user (as shown in Figures 3.6 and 3.7 for the questions *How tall is Stefano Tacconi?*, and *List the children of Margaret Thatcher*, respectively). Each node has an associated confidence score, resulting from the fuzzy labeling algorithm. Moreover, each node is related to the others by a relation of support or attack, and a further specification of such relations according to the categories previously described is provided to the user as answer justification of why the information items have been reconciled and ranked in this way.

Looking at these examples, the reader may argue that the former question can be answered by a simple majority voting (Figure 3.6), and the latter can be answered by a grouping based on surface forms (Figure 3.7), without the need to introduce the complexity of the argumentation machinery. However, if we consider the following example from our dataset, the advantage of using argumentation theory becomes clear. Let us consider the question *Who developed Skype?*: in this case, we retrieve three different answers, namely Microsoft (from FR DBpedia), Microsoft Skype Division (from FR DBpedia), and Skype Limited (EN DBpedia). The relations assigned by RADAR are visualized in Figure 3.8. The answer, with the associated weights, returns first Microsoft (FR) with a confidence score of 0.74, and second, Skype Limited (EN, FR) with a confidence score of 0.61. Note that this result cannot be achieved with simple majority voting nor with grouping based on surface forms.

QA experimental setting

To provide a quantitative evaluation of RADAR integration into QAKiS on a standard dataset of natural language questions, we consider the questions provided by the organizers of QALD-5 for the DBpedia track (the most recent edition of the challenge at the time of this work).[15] More specifically, we collect the questions sets of QALD-2 (i.e. 100 questions of the training and 100 questions of the test sets), the test set of QALD-4 (i.e. 50 questions), and the questions sets of QALD-5 (50 additional training questions with respect to the previous years training

[15] http://www.sc.cit-ec.uni-bielefeld.de/qald/

set, and 59 questions in the test sets). These 359 questions correspond to all the questions released in the five years of the QALD challenge (given the fact that the questions of QALD-1 are included into the question set of QALD-2, and the question set of QALD-3 is the same as QALD-2, but translated into 6 languages, and the training sets of QALD-4 and 5 include all the questions of QALD-2). QALD-3 also provides natural language questions for Spanish DBpedia, but given that the current version of QAKiS cannot query the Spanish DBpedia, we could not use this question set.

We extract from this reference dataset of 359 questions, the questions that the current version of QAKiS is built to address (i.e. questions containing a NE related to the answer through one property of the ontology), corresponding to 26 questions in QALD-2 training set, 32 questions in QALD-2 test sets, 12 in QALD-4 test set, 18 in QALD-5 training set, and 11 in QALD-5 test set. The discarded questions require either some form of aggregation (e.g., counting or ordering), information from datasets different than DBpedia, involve n-ary relations, or are boolean questions. We consider these 99 questions as the QALD reference dataset for our experiments.

Results on QALD answers reconciliation

We run the questions contained into our QALD reference dataset on the English, German, French and Italian chapters of DBpedia. Since the questions of QALD were created to query the English chapter of DBpedia only, it turned out that only in 43/99 cases at least two endpoints provide an answer (in all the other cases the answer is provided by the English chapter only, not useful for our purposes). For instance, given the question *Who developed Skype?* the English DBpedia provides *Skype Limited* as the answer, while the French one outputs *Microsoft* and *Microsoft Skype Division*. Or given the question *How many employees does IBM have?*, the English and the German DBpedia chapters provide 426751 as answer, while the French DBpedia 433362. Table 3.6 lists these 43 QALD questions, specifying which DBpedia chapters (among the English, German, French and Italian ones) contain at least one value for the queried relation. This list of question is the reference question set for our evaluation.

We evaluated the ability of RADAR 2.0 to correctly classify the relations among the information items provided by the different language-specific SPARQL endpoints as answer to the same query, w.r.t. a manually annotated gold standard, built following the methodology in Cabrio et al. [95]. More specifically, we evaluate RADAR with two sets of experiments: in the first case, we start from the answers provided by the different DBpedia endpoints to the 43 QALD questions, and we run RADAR on it. In the second case, we add QAKiS in the loop, meaning that the data we use as input for the argumentation module are directly produced by the system. In this second case, the input are the 43 natural language questions.

Table 3.5 reports on the results we obtained for the two experiments. As already noticed before, the QALD dataset was created to query the English chapter of DBpedia only, and therefore this small dataset does not capture the variability of possibly inconsistent answers that can be found among DBpedia language-specific chapters. Only three categories of relations are present in this data – *surface forms*, *geo-specification*, and *inclusion* – and for this reason RADAR has outstanding performances on it when applied on the correct mapping between NL questions and the SPARQL queries. When QAKiS is added into the loop, its mistakes in interpreting the NL question and translating it into the correct SPARQL query are propagated in RADAR (that receives in those cases a wrong input), decreasing the total performances.

Notice that in some cases the question interpretation can be tricky, and can somehow bias the evaluation of the answers provided by the system. For instance, for the question *Which pope succeeded John Paul II?*, the English DBpedia provides *Benedict XVI* as the answer, while the Italian DBpedia provides also other names of people that were successors of John Paul II in

other roles, as for instance in being the Archbishop of Krakow. But since in the gold standard this question is interpreted as being the successor of John Paul II in the role of Pope, only the entity *Benedict XVI* is accepted as correct answer.

When integrated into QAKiS, RADAR 2.0 outperforms the results obtained by a preliminary version of the argumentation module, i.e. RADAR 1.0 [86], for the positive relation classification (the results of the argumentation module only cannot be strictly compared with the results obtained by RADAR 2.0, since *i)* in its previous version the relation categories are different and less fine-grained, and *ii)* in [86] only questions from QALD-2 were used in the evaluation), showing an increased precision and robustness of our framework. Note that this evaluation is not meant to show that QAKiS performance is improved by RADAR. Actually, RADAR does not affect the capacity of QAKiS to answer questions: RADAR is used to disambiguate among multiple answers retrieved by QAKiS in order to provide to the user the most reliable (and hopefully correct) one.

One of the reasons why RADAR is implemented as a framework that can be integrated on top of an existing QA system architecture (and is therefore system-independent), is because we would like it to be tested and exploited by potentially all QA systems querying more than one DBpedia chapter (at the time of this study, QAKiS was the only one, but given the potential increase in the coverage of a QA system querying multiple DBpedia language-specific chapters [84], we expected other systems to take advantage of these interconnected resources).

Table 3.5: Results on QALD relation classification

System	*Relation category*	*Precision*	*Recall*	F_1
RADAR 2.0 (only)	*surface form*	1.00	0.98	0.99
	geo-specification	0.88	0.80	0.84
	inclusion	0.80	1.00	0.88
	overall positive	1.00	0.98	0.99
baseline	*surface form*	1.00	0.97	0.98
	geo-specification	0.00	0.00	0.00
	inclusion	0.00	0.00	0.00
	overall positive	1.00	0.86	0.92
QAKiS + RADAR 2.0	*surface form*	1.00	0.59	0.74
	geo-specification	0.88	0.80	0.84
	inclusion	0.80	1.00	0.88
	overall positive	1.00	0.63	0.77
QAKiS + baseline	*surface form*	1.00	0.58	0.74
	geo-specification	0.00	0.00	0.00
	inclusion	0.00	0.00	0.00
	overall positive	1.00	0.52	0.68
QAKiS + RADAR 1.0 [86] (on QALD-2 questions only)	**overall positive**	0.54	0.56	0.55

Table 3.6: QALD questions used in the evaluation (in bold the ones correctly answered by QAKiS; *x* means that the corresponding language specific DBpedia chapter (EN, FR, DE, IT) contains at least one value for the queried relation; *dbo* means DBpedia ontology)

ID, question set	*Question*	*DBpedia relation*	*EN*	*FR*	*DE*	*IT*
84, QALD-2 train	Give me all movies with Tom Cruise.	`starring`	x	x	x	
10, QALD-2 train	**In which country does the Nile start?**	`sourceCountry`	x	x		
63, QALD-2 train	**Give me all actors starring in Batman Begins.**	`starring`	x	x	x	x
43, QALD-2 train	Who is the mayor of New York City?	`leaderName`	x		x	x
54, QALD-2 train	Who was the wife of U.S. president Lincoln?	`spouse`	x	x		
6, QALD-2 train	**Where did Abraham Lincoln die?**	`deathPlace`	x	x	x	
31, QALD-2 train	**What is the currency of the Czech Republic?**	`currency`	x	x	x	x
73, QALD-2 train	**Who owns Aldi?**	`keyPerson`	x	x		x
20, QALD-2 train	**How many employees does IBM have?**	`numberOfEmployees`	x	x	x	x
33, QALD-2 train	**What is the area code of Berlin?**	`areaCode`	x			
2, QALD-2 test	**Who was the successor of John F. Kennedy?**	`successor`	x	x		
4, QALD-2 test	How many students does the Free University in Amsterdam have?	`numberOfStudents`	x	x	x	
14, QALD-2 test	Give me all members of Prodigy.	`bandMember`	x	x		
20, QALD-2 test	**How tall is Michael Jordan?**	`height`	x		x	x
21, QALD-2 test	**What is the capital of Canada?**	`capital`	x	x	x	x
35, QALD-2 test	**Who developed Skype?**	`product`	x	x		
38, QALD-2 test	How many inhabitants does Maribor have?	`populationTotal`	x			x
41, QALD-2 test	**Who founded Intel?**	`foundedBy`	x	x		x
65, QALD-2 test	**Which instruments did John Lennon play?**	`instrument`	x	x		
68, QALD-2 test	**How many employees does Google have?**	`numberOfEmployees`	x	x		x
74, QALD-2 test	**When did Michael Jackson die?**	`deathDate`	x	x	x	
76, QALD-2 test	**List the children of Margaret Thatcher.**	`child`	x	x		
83, QALD-2 test	How high is the Mount Everest?	`elevation`	x	x		x
86, QALD-2 test	**What is the largest city in Australia?**	`largestCity`	x	x		
87, QALD-2 test	Who composed the music for Harold and Maude?	`musicComposer`	x		x	x
34, QALD-4 test	**Who was the first to climb Mount Everest?**	`firstAscentPerson`	x		x	
21, QALD-4 test	**Where was Bach born?**	`birthPlace`	x	x	x	x
32, QALD-4 test	**In which countries can you pay using the West African CFA franc?**	`currency`	x		x	
12, QALD-4 test	How many pages does War and Peace have?	`numberOfPages`	x	x		
36, QALD-4 test	Which pope succeeded John Paul II?	`successor`	x			x
30, QALD-4 test	When is Halloween?	`date`	x	x		
259, QALD-5 train	Who wrote The Hunger Games?	`author`	x	x		
280, QALD-5 train	**What is the total population of Melbourne, Florida?**	`populationTotal`	x	x		x
282, QALD-5 train	In which year was Rachel Stevens born?	`birthYear`	x	x	x	x
283, QALD-5 train	**Where was JFK assassinated?**	`deathPlace`	x	x	x	x
291, QALD-5 train	**Who was influenced by Socrates?**	`influencedBy`	x	x		
295, QALD-5 train	Who was married to president Chirac?	`spouse`	x	x		
298, QALD-5 train	**Where did Hillel Slovak die?**	`deathPlace`	x	x	x	x
7, QALD-5 test	Which programming languages were influenced by Perl?	`influencedBy`	x	x	x	x
18, QALD-5 test	**Who is the manager of Real Madrid?**	`manager`	x	x		
19, QALD-5 test	**Give me the currency of China.**	`country`	x		x	
32, QALD-5 test	What does the abbreviation FIFA stand for?	`name`	x		x	x
47, QALD-5 test	**Who were the parents of Queen Victoria?**	`parent`	x		x	x

3.3 Multimedia answer visualization in QAKiS

Given that an increasingly huge amount of multimedia content is now available on the web on almost any topic, we judged it to be extremely interesting to consider them in the QA scenario, in which the best answers may be a combination of text and other media answers [227]. For this reason, we proposed an additional extension of QAKiS that allows to exploit the structured data and metadata describing multimedia content on the linked data to provide a richer and more complete answer to the user, combining textual information with other media content. A first step in this direction consists in determining the best sources and media (image, audio, video, or a hybrid) to answer a query. For this reason, we have carried out an analysis of the questions provided by the QALD challenge, and we have categorized them according to the possible improved multimedia answer visualization. Then, we have extended QAKiS output to

Multimedia	Example question
Picture	*Give me all female Russian astronauts.*
Picture + video	*Give me all movies directed by Francis Ford Coppola.*
Picture + map	*In which country does the Nile start?*
Map + barchart	*Give me all world heritage sites designated within the past 5 years.*
Statcharts	*What is the total amount of men and women serving in the FDNY?*
Timelines	*When was Alberta admitted as province?*

Table 3.7: QALD-3 questions improved answer visualization

include *i)* pictures from Wikipedia Infoboxes, for instance to visualize images of people or places (for questions as *Who is the President of the United States?*); *ii)* OpenStreetMap, to visualize maps for questions asking about a place (e.g. *What is the largest city in Australia?*) and *iii)* YouTube, to visualize videos related to the answer (e.g. a trailer of a movie, for questions like *Which films starring Clint Eastwood did he direct himself?*).

While providing the textual answer to the user, the *multimedia answer generator module* (see Figure 3.2) queries again DBpedia to retrieve additional information about the entity contained in the answer. To display the images, it extracts the properties `foaf:depiction` and `dbpedia-owl:thumbnail`, and their value (i.e. the image) is shown as output. To display the maps (e.g. when the answer is a place), it retrieves the GPS co-ordinates from DBpedia (properties `geo:geometry`, `geo:lat` and `geo:long`), and it injects them dynamically into OpenStreetMap[16] to display the map. Given the fact that DBpedia data can be inconsistent or incomplete, we define a set of heuristics to extract the co-ordinates: in case there are several values for the latitude and longitude, *i)* we give priorities to negative values (indicating the southern hemisphere[17]), and *ii)* we take the value with the highest number of decimal values, assuming it is the most precise. Finally, to embed YouTube[18] videos, first the Freebase[19] ID of the entity is retrieved through the DBpedia property `owl:sameAs`. Then, such ID is used via YouTube search API (v3) (i.e. it is included in the embed code style <`iframe`>, that allows users to view the embedded video in either Flash or HTML5 players, depending on their viewing environment and preferences). Moreover, since we want to have pertinent videos (i.e. showing content related to the answer in the context of the question only), we remove stopwords from the input question, and we send the remaining words as search parameters. For instance, for the question *Give me the actors starring in Batman Begins*, the words "actors", "starring", "Batman Begins" are concatenated and used as search parameters, so that the videos extracted for such actors are connected to the topic of the question (i.e. the actors in their respective roles in Batman Begins).

Figure 3.9 shows QAKiS the above described extension in demo interface. The user can select the DBpedia chapter she wants to query besides English, i.e. French or German DBpedia. Then the user can either write a question or select among a list of examples, and click on *Get Answers!*. As output, in the tab *Results* QAKiS provides: *i)* the textual answer (linked to its DBpedia page), *ii)* the DBpedia source, *iii)* the associate image in Wikipedia Infobox, *iv)* a *more details* button. Clicking on that button, both the entity abstract in Wikipedia, the map and the retrieved videos (if pertinent) are shown.

In order to determine the best sources and media (image, audio, video, or a hybrid) to answer a query, we have carried out an analysis on a subset of the questions provided by the QALD-3

[16] `www.openstreetmap.org`

[17] We verified that when both a positive and a negative value are proposed, the negative is the correct one (the letter S, i.e. South, is not correctly processed.)

[18] `www.youtube.com/`

[19] `www.freebase.com`

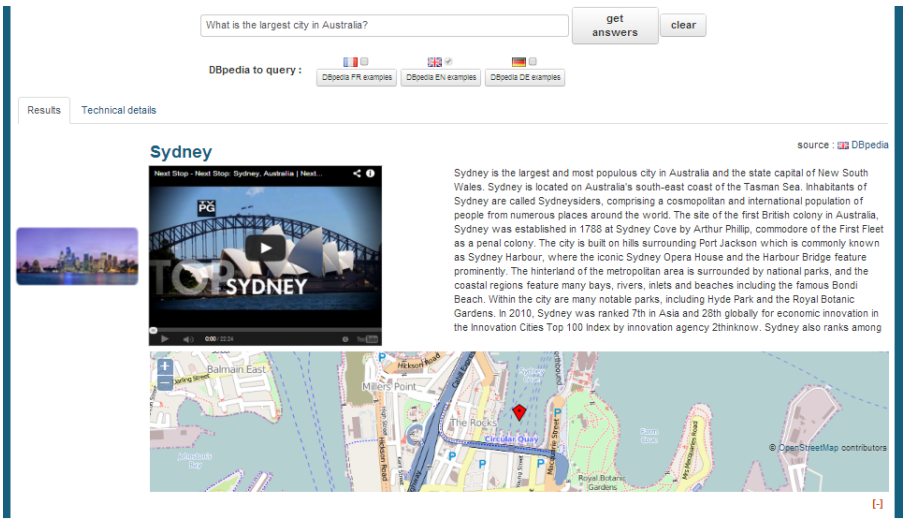


Figure 3.9: QAKiS with multimedia visualization

challenge.[20] The goal was to categorize them according to the possible improved multimedia answer visualization, and to extract some heuristics to be exploited by QAKiS to provide the most complete answer to a certain question. In this analysis, we discarded the questions for which no additional multimedia content would be pertinent, e.g. questions whose answer is a number (e.g. *How many students does the Free University in Amsterdam have?*), or boolean questions (e.g. *Did Tesla win a nobel prize in physics?*). In future work we could provide multimedia content on the entity in the question, but in the current work we are focusing on boosting the answer visualization only. Table 3.7 shows the categories of multimedia content for answer visualization on which we are focusing, together with an example of question for which such kind of multimedia content would be appropriate.

3.4 Related Work

QA systems over Linked Data. The most recent survey on the field of Question Answering at the time of the works presented in this chapter is provided by [290], with a focus on ontology-based QA. Moreover, they examine the potential of open user-friendly interfaces for the SW to support end users in reusing and querying SW content. State of the art QA systems over Linked Data generally address the issue of question interpretation mapping a natural language question to a triple-based representation. For instance, Freya [134] uses syntactic parsing in combination with ontology-based lookup for question interpretation, partly relying on the user's help in selecting the entity that is most appropriate as match for some natural language expressions. One of the problems of that approach is that often end-users are unable to help, in case they are not informed about the modeling and vocabulary of the data. PowerAqua [289] accepts user queries expressed in NL and retrieves answers from multiple semantic sources on the SW. It follows a pipeline architecture, according to which the question is *i)* transformed by the linguistic component into a triple based intermediate format, *ii)* passed to a set of components to identify potentially suitable semantic entities in various ontologies, and then *iii)* the various interpretations produced in different ontologies are merged and ranked for answer retrieval. PowerAqua's main limitation is in its linguistic coverage. Pythia [446] relies on a deep linguistic analysis to compositionally construct meaning representations using a vocabulary aligned to the vocabulary of a given ontology. Pythia's major drawback is that it requires a lexicon, which has to be manually created. Later, an approach based on Pythia [446] but more

[20] http://doi.org/10.4119/unibi/citec.2013.6

similar to the one adopted in QAKiS is presented [445]. It relies on a linguistic parse of the question to produce a SPARQL template that directly mirrors the internal structure of the question (i.e. SPARQL templates with slots to be filled with URIs). This template is then instantiated using statistical entity identification and predicate detection.

Information reconciliation State-of-the-art QA systems over Linked Data generally address the issue of question interpretation mapping a natural language question to a triple-based representation (see [290] for an overview). Moreover, they examine the potential of open user-friendly interfaces for the Semantic Web to support end users in reusing and querying the Semantic Web content. None of these systems considers language-specific DBpedia chapters, and they do not provide a mechanism to reconcile the different answers returned by heterogeneous endpoints. Finally, none of them provides explanations about the answer returned to the user.

Several works address alignment agreement based on argumentation theory. More precisely, Laera et al. [261] address alignment agreement relying on argumentation to deal with the arguments which attack or support the candidate correspondences among ontologies. Doran et al. [147] propose a methodology to identify subparts of ontologies which are evaluated as sufficient for reaching an agreement, before the argumentation step takes place, and dos Santos and Euzenat [148] present a model for detecting inconsistencies in the selected sets of correspondences relating ontologies. In particular, the model detects logical and argumentation inconsistency to avoid inconsistencies in the agreed alignment. We share with these approaches the use of argumentation to detect inconsistencies, but RADAR goes beyond them: we identify in an automated way relations among information items that are more complex than `owl:sameAs` links (as in ontology alignment). Moreover, these approaches do not consider trust-based acceptance degrees of the arguments, lacking to take into account a fundamental component in the arguments' evaluation, namely their sources.

We mentioned these works applying argumentation theory to address ontology alignment agreements as examples of applications of this theory to open problems in the Semantic Web domain. Actually, the two performances cannot be compared to show the superiority of one of the two approaches, as the task is different.

QALD competitions I was involved in the organization of four editions of the Question Answering over Linked Data challenge (i.e. QALD-3,4,5 and 6), a series of evaluation campaigns on question answering over linked data with a strong emphasis on multilingualism, hybrid approaches using information from both structured and unstructured data, and question answering over RDF data cubes. It is aimed at all kinds of systems that mediate between a user, expressing his or her information need in natural language, and semantic data. The main objective of QALD is to provide up-to-date, demanding benchmarks that establish a standard against which question answering systems over structured data can be evaluated and compared. It has been organized as an ESWC workshop and an ISWC workshop as well as a part of the Question Answering lab at CLEF. Over the past years, a range of approaches have been developed to address this challenge, showing significant advances towards answering natural language questions with respect to large, heterogeneous sets of structured (and unstructured data). In [114, 447, 448, 450] we described the computational tasks of the different editions of the competition, the technical setup, and presented the participants. Furthermore, we reported on the obtained results and provided some analysis and interpretation.

3.5 Conclusions

The work presented in this chapter is interdisciplinary with respect to the research fields of Natural Language Processing and the Semantic Web, to enhance interactions between non-expert users and the huge and heterogeneous amount of data available on the Web. In particular, we have presented QAKiS, a QA system over DBpedia that allows end users to submit a query to an RDF triple store in English and obtain the answer in the same language, hiding the complexity of the non-intuitive formal query languages involved in the resolution process. To show the interesting potential for NLP applications resulting from the properties alignment in multilingual DBpedia, we have extended the QAKiS system so that it could query the ontology properties of the French, German and Italian DBpedia chapters. We show that this integration extends the system coverage (i.e. the recall), without having a negative impact on its precision. With this respect, possible extensions of the proposed framework are envisaged, to improve mainly: i) the system coverage, addressing boolean and n-relation questions; ii) port QAKiS to multiple languages, i.e. allowing input questions in languages different from English.

We have also presented another extension of QAKiS, that allows to complement textual answers with multimedia content from the linked data, to provide a richer and more complete answer to the user.

Finally, a major contribution of this chapter is the introduction and evaluation of the RADAR 2.0 framework for information reconciliation over language-specific DBpedia chapters. The framework is composed of three main modules: a module computing the confidence score of the sources depending either on the length of the related Wikipedia page or on the geographical characterization of the queried entity, a module retrieving the relations holding among the elements of the result set, and finally a module computing the reliability degree of such elements depending on the confidence assigned to the sources and the relations among them. This third module is based on bipolar argumentation theory, and a bipolar fuzzy labeling algorithm is exploited to return the acceptability degrees. The resulting graph of the result set, together with the acceptability degrees assigned to each information item, justifies to the user the returned answer and it is the result of the reconciliation process. The evaluation of the framework shows the feasibility of the proposed approach. Moreover, the framework has been integrated in QAKiS, allowing to reconcile and justify the answers obtained from four language-specific DBpedia chapters (i.e. English, French, German and Italian). Finally, the resource generated applying RADAR to 300 properties in 15 DBpedia chapters to reconcile their values is released. There are several points to be addressed as future work, as the user evaluation, to verify whether our answer justification in QAKiS appropriately suits the needs of the data consumers, and to receive feedback on how to improve such visualization. Moreover, the proposed framework is not limited to the case of multilingual chapters of DBpedia. The general approach RADAR is based on allows to extend it to various cases like inconsistent information from multiple English data endpoints. The general framework would be the same, the only part to be defined are the rules to extract the relations among the retrieved results. Investigating how a module of this type can be adopted as a fact-checking module is part of our future research plan.

Chapter 4

Mining argumentative structures from texts

This section is dedicated to my research about argument mining, a research area I contributed to raise together with Serena Villata [92]. My research in this area mainly deals with the detection of arguments and the prediction of their relations in different textual resources, as political debates, medical texts, and social media content.

My research contributions on this topic have been published in several journal and venues. I provide below a list of the main publications on the topic:

- Shohreh Haddadan, Elena Cabrio and Serena Villata (2019). *Yes, we can! Mining Arguments in 50 Years of US Presidential Campaign Debates*, in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL2019). Short paper [209]

- Tobias Mayer, Elena Cabrio, Serena Villata (2020). *Transformer-based Argument Mining for Healthcare Applications*, Proceedings of the 24th European Conference on Artificial Intelligence (ECAI2020) [304]

- Tobias Mayer, Elena Cabrio, Serena Villata (2019). ACTA A Tool for Argumentative Clinical Trial Analysis. Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI 2019). Demo paper [303]

- Shohreh Haddadan, Elena Cabrio, Serena Villata (2019) DISPUTool - A tool for the Argumentative Analysis of Political Debates. Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI 2019). Demo paper [208]

- Stefano Menini, Elena Cabrio, Sara Tonelli and Serena Villata (2018). *Never retreat, never retract: Argumentation Analysis for Political Speeches*, Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18) [312]

- Elena Cabrio, Serena Villata (2018), *Five Years of Argument Mining: a Data-driven Analysis*, Proceedings of IJCAI-ECAI 2018 [92]

- Mihai Dusmanu, Elena Cabrio and Serena Villata (2017). *Argument Mining on Twitter: Arguments, Facts and Sources*, in Proceedings of the Empirical Methods in Natural Language Processing conference (EMNLP2017). Short paper [154]

- Sahbi Benlamine, Maher Chaouachi, Serena Villata, Elena Cabrio, Claude Frasson, Fabien Gandon (2015). *Emotions in Argumentation: an Empirical Evaluation*, Proceedings of the International Joint Conference in Artificial Intelligence (IJCAI 2015) [48]

- Elena Cabrio, Serena Villata (2013). A natural language bipolar argumentation approach to support users in online debate interactions†. Argument & Computation 4(3): 209-230 (2013) [91]

- Elena Cabrio, Serena Villata and Fabien Gandon (2013). *A Support Framework for Argumentative Discussions Management in the Web.* in The Semantic Web: Semantics and Big Data, Lecture Notes in Computer Science Volume 7882, pp. 412-426. **Best paper award** [94]

- Elena Cabrio, Serena Villata (2012). *Natural Language Arguments: A Combined Approach*, in Proceedings of the European Conference on Artificial Intelligence (ECAI 2012) [90]

- Elena Cabrio, Serena Villata (2012). *Combining Textual Entailment and Argumentation Theory for Supporting Online Debates Interactions*, Proceedings of the 50th annual meeting of the Association for Computational Linguistics (ACL 2012). Short paper. [89]

AM is "the general task of analyzing discourse on the pragmatics level and applying a certain argumentation theory to model and automatically analyze the data at hand" [206]. Two tasks are crucial in Argument Mining: *i) Argument component detection* in the input text: this step may be further split in the detection of argument components (i.e., claims and premises) and of their textual boundaries. Different methods have been tested in the last years, like Support Vector Machines (SVM) (e.g., [280]), Naïve Bayes classifiers [156], Logistic Regression [274] and Neural Networks [425], and *ii) Prediction of the relations* holding between the argumentative components (i.e., *attacks* and *supports*). Relations can be predicted between arguments [91] or between argument components [425].

In this chapter, I summarize my main contributions in the AM research field, from my very first works on bridging semantic inferences methods in NLP with bipolar argumentation to analyze user-generated content, to my ongoing research on defining and applying AM methods in the political and in the medical scenarios. The contributions reported in this chapter are the results of several collaborations with my colleague Serena Villata, in the context of the Ph.D. of Tobias Mayer and Shohreh Haddadan.

Our very first works on user-generated content raised from the observation that with the growing use of the Social Web, an increasing number of applications for exchanging opinions with other people are becoming available online. These applications are widely adopted with the consequence that the number of opinions about the debated issues increases. In order to cut in on a debate, the participants need first to evaluate the opinions of the other users to detect whether they are in favour or against the debated issue. Bipolar argumentation proposes algorithms and semantics to evaluate the set of accepted arguments, given the support and the attack relations among them. Two main problems arise. First, an automated framework to detect the relations among the arguments represented by the natural language (NL) formulation of the users' opinions is needed. Together with my colleague Serena Villata, we addressed this open issue by proposing and evaluating the use of NL techniques to identify the arguments and their relations. In particular, we adopt the Textual Entailment (TE) approach, a generic framework for applied semantics, where linguistic objects are mapped by means of semantic inferences at a textual level. TE is then coupled together with an abstract bipolar argumentation system which allows to identify the arguments that are accepted in the considered online debate. Second, we address the problem of studying and comparing the different proposals put forward for modelling the support relation. The emerging scenario shows that there is not a unique interpretation of the support relation. In particular, different combinations of additional attacks among the arguments involved in a support relation are proposed. We provide an NL account of the notion of support based on online debates, by discussing and evaluating the

support relation among arguments with respect to the more specific notion of TE in the NL processing field. Finally, we carry out a comparative evaluation of four proposals of additional attacks on a sample of NL arguments extracted from Debatepedia. The originality of the proposed framework lies in the following point: NL debates are analysed and the relations among the arguments are automatically extracted.

In another work on user-generated content, we addressed the problem of understanding the stream of messages exchanged on social media such as Facebook and Twitter, given that is becoming a major challenge for automated systems. The tremendous amount of data exchanged on these platforms as well as the specific form of language adopted by social media users constitute a new challenging context for existing argument mining techniques. We constructed a resource of natural language arguments called DART (Dataset of Arguments and their Relations on Twitter) [68] where the complete argument mining pipeline over Twitter messages is considered: (i) we identify which tweets can be considered as arguments and which cannot, and (ii) we identify what is the relation, i.e., support or attack, linking such tweets to each other. We also worked on the creation of a complete argument mining pipeline over Twitter messages [69], whose final goal is to compute the set of tweets which are widely recognized as accepted, and the different (possibly conflicting) viewpoints that emerge on a topic, given a stream of messages. In addition, new issues emerge when dealing with arguments posted on such platforms, such as the need to make a distinction between personal opinions and actual facts, and to detect the source disseminating information about such facts to allow for provenance verification. We applied supervised classification to identify arguments on Twitter, and we presented two new tasks for argument mining, namely facts recognition and source identification. We studied the feasibility of the approaches proposed to address these tasks on a set of tweets related to the Grexit and Brexit news topics.

One of the main researches we are currently focusing on concern AM on political speeches, with the final goal of addressing fallacies detection. As a first work in this area, we applied argumentation mining techniques, in particular relation prediction, to study political speeches in monological form, where there is no direct interaction between opponents. We argue that this kind of technique can effectively support researchers in history, social and political sciences, which must deal with an increasing amount of data in digital form and need ways to automatically extract and analyze argumentation patterns. We tested and discussed our approach based on the analysis of documents issued by R. Nixon and J.F. Kennedy during 1960 presidential campaign. The application of argument mining to such data allows not only to highlight the main points of agreement and disagreement between the candidates' arguments over the campaign issues such as Cuba, disarmament and health-care, but also an in-depth argumentative analysis of the respective viewpoints on these topics.

Given that political debates offer a rare opportunity for citizens to compare the candidates' positions on the most controversial topics of the campaign, as a second line of works in this area, we carried out an empirical investigation of the typology of argument components in political debates by annotating 39 political debates from the last 50 years of US presidential campaigns, creating a new corpus of 29k argument components, labeled as premises and claims. Moreover, we evaluated feature-rich SVM learners and Neural Networks methods on such data achieving satisfactory performances on different oratory styles across time and topics.

As for the medical domain, evidence-based decision making in the health-care domain targets at supporting clinicians in their deliberation process to establish the best course of action for the case under evaluation. Although the reasoning stage of this kind of frameworks received considerable attention, little effort has been devoted to the mining stage. In this research activity, we annotated first a dataset of 159 abstracts of Randomized Controlled Trials (RCTs) from the MEDLINE database, comprising 4 different diseases (i.e., glaucoma, hypertension, hepatitis b, diabetes), then a larger dataset of 500 abstracts on the *neoplasm* disease, leading

to a dataset of 4113 argument components and 2601 argument relations. We then proposed a complete argument mining pipeline for RCTs, classifying argument components as *evidence* and *claims*, and predicting the relation, i.e., *attack* or *support*, holding between those argument components. We experiment with deep bidirectional transformers in combination with different neural architectures (i.e., LSTM, GRU and CRF) and outperformed current state-of-the-art end-to-end argument mining systems.

This chapter is organized as follows: Section 4.1 and 4.2 present the research work carried out on user-generated content. Section 4.3 and 4.4 report on the work carried out on political speeches and debates, while Section 4.5 describes the recent work on AM on clinical trials. We report then related work on AM, and conclusions end the chapter.

4.1 A natural language bipolar argumentation approach to support users in online debate interactions

In the last years, the Web has changed in the so called Social Web. The Social Web has seen an increasing number of applications like Twitter[1], Debatepedia[2], Facebook[3] and many others, which allow people to express their opinions about different issues. Let us consider for instance the following debate published on Debatepedia: the issue of the debate is "Making Internet a right only benefits society". The participants have proposed various pro and con arguments concerning this issue, e.g., a pro argument claims that the Internet delivers freedom of speech, and a con argument claims that the Internet is not as important as real rights like the freedom from slavery. These kinds of debates are composed by tens of arguments in favour or against a proposed issue. The main difficulty for newcomers is to understand the current holding position in the debate, i.e., to understand which are the arguments that are accepted at a certain moment. This difficulty is twofold: first, the participants have to remember all the different, possibly long, arguments and understand which are the relations among these arguments, and second they have to understand, given these relations, which are the accepted arguments.

In this work, one of our first work on AM, we answer the following research question: *how to support the participants in natural language (NL) debates to detect which are the relations among the arguments, and which arguments are accepted?* Two kinds of relations connect the arguments in such online debate platforms: a positive relation (i.e., a *support* relation), and a negative relation (i.e., an *attack* relation). To answer to our research question we need to rely on an argumentative framework able to deal with such *bipolar* relations. [152]'s abstract theory defines an argumentation framework as a set of abstract arguments interacting with each others through a so called *attack* relation. In the last years, several proposals to extend the original abstract theory with a *support* relation have been addressed, leading to the birth of *bipolar argumentation* frameworks (BAF) [100], and the further introduction of a number of *additional attacks* among the arguments [101, 61, 348].

Our research question breaks down into the following subquestions:

1. How to automatically identify the arguments, as well as their relationships, from natural language debates?

2. What is the relation between the notion of support in bipolar argumentation and the notion of textual entailment in natural language processing?

[1] `http://twitter.com/`
[2] `http://idebate.org/`
[3] `http://www.facebook.com/`

First, we propose to combine natural language techniques and Dung-like abstract argumentation to identify and generate the arguments from natural language text, and then to evaluate this set of arguments to know which are the accepted ones. Starting from the participants' opinions, we detect which ones imply or contradict, even indirectly, the issue of the debate using the textual entailment approach. Beside formal approaches to semantic inference that rely on logical representation of meaning, the notion of Textual Entailment (TE) has been proposed as an applied framework to capture major semantic inference needs across applications in the Computational Linguistics field [132]. The development of the Web has witnessed a paradigm shift, due to the need to process a huge amount of available (but often noisy) data. TE is a generic framework for applied semantics, where linguistic objects are mapped by means of semantic inferences at a textual level. We use TE to automatically identify, from a natural language text, the arguments. Second, we adopt bipolar argumentation [100] to reason over the set of generated arguments with the aim of deciding which are the accepted ones. Proposals like argumentation schemes [465], Araucaria [391], Carneades [196], and ArguMed [455] use natural language arguments, but they ask the participants to indicate the semantic relationship among the arguments, and the linguistic content remains unanalyzed. As underlined by [390], "the goal machinery that leads to arguments being automatically generated has been only briefly touched upon, and yet is clearly fundamental to the endeavor". Summarizing, we combine the two approaches, i.e., textual entailment and abstract bipolar argumentation, in a framework whose aim is to *(i)* generate the abstract arguments from the online debates through TE, *(ii)* build the argumentation framework from the arguments and the relationships returned by the TE module, and *(iii)* return the set of accepted arguments. We evaluate the feasibility of our combined approach on a data set extracted from a sample of Debatepedia debates.

Second, we study the relation among the notion of support in bipolar argumentation [100], and the notion of TE in Natural Language Processing (NLP) [132]. In the first study of the current work, we assume the TE relation extracted from NL texts as equivalent to a support relation in bipolar argumentation. This is a strong assumption, and in this second part of our work we aim at verifying on a sample of real data from Debatepedia whether it is always the case that support is equivalent to TE. In particular, for addressing this issue we focus both on the relation between support and entailment, and on the relation between attack and contradiction. We show that TE and contradiction are more specific concepts than support and attack, but still hold in most of the argument pairs. Moreover, starting from the comparative study addressed by [102], we consider four additional attacks proposed in the literature: *supported* (if argument a supports argument b and b attacks argument c, then a attacks c) and *secondary* (if a supports b and c attacks a, then c attacks b) attacks [101], *mediated* attacks [61] (if a supports b and c attacks b, then c attacks a), and *extended* attacks [347, 348] (if a supports b and a attacks c, then b attacks c). We investigate the presence and the distribution of these attacks in NL debates on a data set extracted from Debatepedia, and we show that all these models are verified in human debates, even if with a different frequency.

The originality of the proposed framework consists in the combination of two techniques which need each other to provide a complete reasoning model: TE has the power to automatically identify the arguments in the text and to specify which kind of relation links each couple of arguments, but it cannot assess which are the *winning* arguments. This is addressed by argumentation theory which lacks automatic techniques to extract the arguments from free text. The combination of these two approaches leads to the definition of a powerful tool to reason over online debates. In addition, the benefit of the proposed deeper analysis of the relation among the two notions of support and TE is twofold. First, it is used to verify, through a data driven evaluation, the "goodness" of the proposed models of bipolar argumentation to be used in real settings, going beyond *ad hoc* NL examples. Second, it can be used to guide the construction of cognitive agents whose major need is to achieve a behavior as close as possible

to the human one.

In the following, we first provide an overview on the standard approaches to semantic inference in the natural language processing field as well as an introduction to textual entailment. We then summarize the basic notions of bipolar argumentation, and describe the four kinds of additional attacks we consider in this work. We present our combined framework unifying textual entailment and bipolar argumentation towards the automated detection of the arguments' relations and their acceptability, and we address the analysis of the meaning of support and attack in natural language dialogues, as well as the comparative study on the existing additional attacks.

4.1.1 NLP approaches to semantic inference

Classical approaches to semantic inference rely on logical representations of meaning that are external to the language itself, and are typically independent of the structure of any particular natural language. Texts are first translated, or interpreted, into some logical form and then new propositions are inferred from interpreted texts by a logical theorem prover. But, especially after the development of the Web, we have witnessed a paradigm shift, due to the need to process a huge amount of available (but often noisy) data. Addressing the inference task by means of logical theorem provers in automated applications aimed at natural language understanding has shown several intrinsic limitations [55]. As highlighted in [335], in formal approaches semanticists generally opt for rich (i.e. including at least first order logic) representation formalisms to capture as many relevant aspects of the meaning as possible, but practicable methods for generating such representations are very rare. The translation of real-world sentences into logic is difficult because of issues such as ambiguity or vagueness [381]. Moreover, the computational costs of deploying first-order logic theorem prover tools in real world situations may be prohibitive, and huge amounts of additional linguistic and background knowledge are required. Formal approaches address forms of deductive reasoning, and therefore often exhibit a too high level of precision and strictness as compared to human judgments, that allow for uncertainties typical of inductive reasoning [67]. While it is possible to model elementary inferences on the precise level allowed by deductive systems, many pragmatic aspects that play a role in everyday inference cannot be accounted for. Inferences that are plausible but not logically stringent cannot be modeled in a straightforward way, but in NLP applications approximate reasoning should be preferred in some cases to having no answers at all.

Especially in data-driven approaches, like the one sought in this work, where patterns are learnt from large-scale naturally-occurring data, we can settle for approximate answers provided by efficient and robust systems, even at the price of logic unsoundness or incompleteness. Starting from these considerations, [335] propose to address the inference task directly at the textual level instead, exploiting currently available NLP techniques. While methods for automated deduction assume that the arguments in input are already expressed in some formal meaning representation (e.g., first order logic), addressing the inference task at a textual level opens different and new challenges from those encountered in formal deduction. Indeed, more emphasis is put on informal reasoning, lexical semantic knowledge, and variability of linguistic expressions.

The notion of Textual Entailment has been proposed as an applied framework to capture major semantic inference needs across applications in NLP [132]. It is defined as a relation between a coherent textual fragment (the Text T) and a language expression, which is considered as the Hypothesis (H). Entailment holds (i.e. $T \Rightarrow H$) if the meaning of H can be inferred from the meaning of T, as interpreted by a typical language user. The TE relationship is directional, since the meaning of one expression may usually entail the other, while the opposite is much less certain. Consider the pairs in Example 1 and 2.

Example 1

T1*: Internet access is essential now; must be a right. The internet is only that wire that delivers freedom of speech, freedom of assembly, and freedom of the press in a single connection.*
H*: Making Internet a right only benefits society.*

Example 2 (Continued)

T2*: Internet not as important as real rights. We may think of such trivial things as a fundamental right, but consider the truly impoverished and what is most important to them. The right to vote, the right to liberty and freedom from slavery or the right to elementary education.*
H*: Making Internet a right only benefits society.*

A system aimed at recognizing TE should detect an inference relation between T1 and H (i.e. the meaning of H can be derived from the meaning of T) in Example 1, while it should not detect an entailment between T2 and H in Example 2. As introduced before, TE definition is based on (and assumes) common human understanding of language, as well as common background knowledge. However, the entailment relation is said to hold only if the statement in the text licenses the statement in the hypothesis, meaning that the content of T and common knowledge together should entail H, and not background knowledge alone. In this applied framework, inferences are performed directly over lexical-syntactic representations of the texts. Such definition of TE captures quite broadly the reasoning about language variability needed by different applications aimed at NL understanding and processing, e.g., information extraction [398] and text summarization [36]. Differently from the classical semantic definition of entailment [109], the notion of TE accounts for some degree of uncertainty allowed in applications (see Example 1).

In 2005, the PASCAL Network of Excellence started an attempt to promote a generic evaluation framework covering semantic-oriented inferences needed for practical applications, launching the Recognizing Textual Entailment challenge [132], with the aim of setting a unifying benchmark for the development and evaluation of methods that typically address similar problems in different, application-oriented, manners. As many of the needs of several NLP applications can be cast in terms of TE, the goal of the evaluation campaign is to promote the development of general entailment recognition engines, designed to provide generic modules across applications. Since 2005, such initiative has been repeated yearly[4], asking the participants to develop a system that, given two text fragments (the *text* T and the *hypothesis* H), can determine whether the meaning of one text is entailed, i.e. can be inferred, from the other. For pairs where the entailment relation does not hold between T and H, systems are required to make a further distinction between pairs where the entailment does not hold because the content of H is contradicted by the content of T (i.e. *contradiction*, see Example 2), and pairs where the entailment cannot be determined because the truth of H cannot be verified on the basis of the content of T (i.e. *unknown*, see Example 3). [300] provide a definition of contradiction for the TE task, claiming that it occurs when two sentences *i)* are extremely unlikely to be true simultaneously, and *ii)* involve the same event. This three-way judgment task (*entailment* vs *contradiction* vs *unknown*) was introduced since RTE-4, while before a two-way decision task (*entailment* vs *no entailment*) was asked to participating systems. However, the classic two-way task is offered as an alternative also in recent editions of the evaluation campaign (*contradiction* and *unknown* judgments are collapsed into the judgment *no entailment*).

In our work, we consider the three way scenario to map TE relation with bipolar argumentation, focusing both on the relation between support and entailment, and on the relation

[4] `http://aclweb.org/aclwiki/index.php?title=Recognizing_Textual_Entailment`

between attack and contradiction. As will be discussed in Section 4.1.5, we consider argument pairs connected by a relation of support (but where the first argument does not entail the second one), and argument pairs connected by a relation of attack (but where the first argument does not contradict the second one) as *unknown* pairs in the TE framework.

Example 3 (Continued)

T3*: Internet "right" means denying parents' ability to set limits. Do you want to make a world where a mother tells her child: "you cannot stay on the internet anymore" that she has taken a right from him? Compare taking the right for a home or for education with taking the "right" to access the internet.*
H*: Internet access is essential now; must be a right. The internet is only that wire that delivers freedom of speech, freedom of assembly, and freedom of the press in a single connection.*

The systems submitted to the RTE challenge are tested against manually annotated data sets, which include typical examples that correspond to success and failure cases of NLP applications. A number of data-driven approaches applied to semantics have been experimented throughout the years. In general, the approaches still more used by the submitted systems include Machine Learning (typically SVM), logical inference, cross-pair similarity measures between T and H, and word alignment - for an overview, see [13], and [132].

4.1.2 Bipolar argumentation

This section provides the basic concepts of [152]'s abstract argumentation, and bipolar argumentation [100].

Definition 3 *(Abstract argumentation framework AF) An abstract argumentation framework is a pair $\langle A, \rightarrow \rangle$ where A is a set of elements called arguments and $\rightarrow \subseteq A \times A$ is a binary relation called attack. We say that an argument a attacks an argument b if and only if $(a, b) \in \rightarrow$.*

[152] presents several acceptability semantics that produce zero, one, or several sets of accepted arguments. Such semantics are grounded on two main concepts called conflict-freeness and defence.

Definition 4 *(Conflict-free, Defence) Let $C \subseteq A$. A set C is conflict-free if and only if there exist no $a, b \in C$ such that $a \rightarrow b$. A set C defends an argument a if and only if for each argument $b \in A$ if b attacks a then there exists $c \in C$ such that c attacks b.*

Definition 5 *(Acceptability semantics) Let C be a conflict-free set of arguments, and let $\mathcal{D} : 2^A \mapsto 2^A$ be a function such that $\mathcal{D}(C) = \{a | C\ defends\ a\}$.*

- *C is admissible if and only if $C \subseteq \mathcal{D}(C)$.*
- *C is a complete extension if and only if $C = \mathcal{D}(C)$.*
- *C is a grounded extension if and only if it is the smallest (w.r.t. set inclusion) complete extension.*
- *C is a preferred extension if and only if it is a maximal (w.r.t. set inclusion) complete extension.*
- *C is a stable extension if and only if it is a preferred extension that attacks all arguments in $A \setminus C$.*

Roughly, an argument is accepted if all its attackers are rejected, and it is rejected if it has at least an attacker which is accepted.

Bipolar argumentation frameworks, firstly proposed by [100], extend Dung's framework taking into account both the attack relation and the support relation. In particular, an abstract bipolar argumentation framework is a labeled directed graph, with two labels indicating either attack or support. In this work, we represent the attack relation by $a \rightarrow b$, and the support relation by $a \Rightarrow b$.

Definition 6 *(Bipolar argumentation framework) A bipolar argumentation framework (BAF) is a tuple* $\langle A, \rightarrow, \Rightarrow \rangle$ *where* A *is the set of elements called arguments, and two binary relations over* A *are called attack and support, respectively.*

[102] address a formal analysis of the models of support in bipolar argumentation to achieve a better understanding of this notion and its uses. [100, 101] argue about the emergence of new kinds of attacks from the interaction between attacks and supports in BAF. In the rest of the section, we will adopt their terminology to refer to additional attacks, i.e., *complex attacks*. In particular, they specify two kinds of complex attacks called *secondary* and *supported* attacks, respectively.

Definition 7 *(Secondary and supported attacks) Let* $BAF = \langle A, \rightarrow, \Rightarrow \rangle$ *where* $a, b \in A$. *A supported attack for* b *by* a *is a sequence* $a_1 R_1 \dots R_{n-1} a_n$, $n \geq 3$, *with* $a_1 = a, a_n = b$, *such that* $\forall i = 1 \dots n-2, R_i \Rightarrow$ *and* $R_{n-1} \rightarrow$. *A secondary attack for* b *by* a *is a sequence* $a_1 R_1 \dots R_{n-1} a_n$, $n \geq 3$, *with* $a_1 = a, a_n = b$, *such that* $R_1 \rightarrow$ *and* $\forall i = 2 \dots n\text{-}1, R_i \Rightarrow$.

According to the above definition, these attacks hold in the first two cases depicted in Figure 4.1, where there is a supported attack from a to c, and there is a secondary attack from c to b. In this work, we represent complex attacks using a dotted arrow.

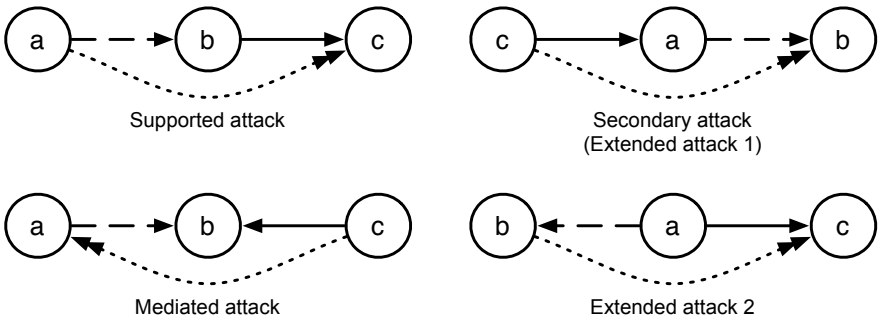


Figure 4.1: Additional attacks emerging from the interaction of supports and attacks.

The support relation has been specialized in other approaches where new complex attacks emerging from the combination of existing attacks and supports are proposed. [61] propose a *deductive* view of support in abstract argumentation where, given the support $a \Rightarrow b$ the acceptance of a implies the acceptance of b, and the rejection of b implies the rejection of a. They introduce a new kind of complex attacks called *mediated* attacks (Figure 4.1).

Definition 8 *(Mediated attacks) Let* $BAF = \langle A, \rightarrow, \Rightarrow \rangle$ *where* $a, b \in A$. *A mediated attack on* b *by* a *is a sequence* $a_1 R_1 \dots R_{n-2} a_{n-1}$ *and* $a_n R_{n-1} a_{n-1}$, $n \geq 3$, *with* $a_1 = a, a_{n-1} = b, a_n = c$, *such that* $R_{n-1} \rightarrow$ *and* $\forall i = 1 \dots n\text{-}2, R_i \Rightarrow$.

[347, 348] propose, instead, an account of support called *necessary* support. In this framework, given $a \Rightarrow b$ then the acceptance of a is necessary to get the acceptance of b, i.e., the

acceptance of b implies the acceptance of a. They introduce two new kinds of complex attacks called *extended attacks* (Figure 4.1). Note that the first kind of extended attacks is equivalent to the secondary attacks introduced by [100, 101], and that the second case is the dual of supported attacks. See [102] for a formal comparison of the different models of support in bipolar argumentation.

Definition 9 *(Extended attacks) Let* $BAF = \langle A, \rightarrow, \Rightarrow \rangle$ *where* $a, b \in A$. *An extended attack on* b *by* a *is a sequence* $a_1 R_1 a_2 R_2 \dots R_n a_n$, $n \geq 3$, *with* $a_1 = a, a_n = b$, *such that* $R_1 = \rightarrow$ *and* $\forall i = 2 \dots n, R_i = \Rightarrow$, *or a sequence* $a_1 R_1 \dots R_n a_n$ *and* $a_1 R_p a_p$, $n \geq 2$, *with* $a_n = a, a_p = b$, *such that* $R_p = \rightarrow$ *and* $\forall i = 1 \dots n, R_i = \Rightarrow$.

All these models of support in bipolar argumentation address the problem of how to compute the set of extensions from the extended framework providing different kinds of solutions, i.e., introducing the notion of *safety* in BAF [100], or computing the extensions in the meta-level [61, 101]. In this work, we are not interested in discussing and evaluating these different solutions. Our aim is to evaluate how much these different models of support occur and are effectively "exploited" in natural language dialogues, towards a better understanding of the notion of support and attack in bipolar argumentation.

We are aware that the notion of support is controversial in the field of argumentation theory. In particular, another view of support sees this relation as a relation holding among the premises and the conclusion of a structured argument, and not as another relation among atomic arguments [385]. However, given the amount of attention bipolar argumentation is receiving in the literature [389], a better account of this kind of frameworks is required.

Another approach to model support has been proposed by [353] and [354], where they distinguish among *prima-facie* arguments and standard ones. They show how a set of arguments described using Dung's argumentation framework can be mapped from and to an argumentation framework that includes both attack and support relations. The idea is that an argument can be accepted only if there is an evidence supporting it, i.e., evidence is represented by means of *prima-facie* arguments. In this section, we do not intend to take a position in this debate. We focus our analysis on the abstract models of bipolar argumentation proposed in the literature [101, 61, 348], and we leave as future work the account of support in structured argumentation and the model proposed by [353] and [354].

4.1.3 Casting bipolar argumentation as a TE problem

The goal of our work is to propose an approach to support the participants in forums or debates (e.g., Debatepedia, Twitter) to detect which arguments among the ones expressed by the other participants on a certain topic are accepted. As a first step, we need to *(i)* automatically generate the arguments (i.e. recognize a participant's opinion on a certain topic as an argument), as well as *(ii)* detect their relation with respect to the other arguments. We cast the described problem as a TE problem, where the T-H pair is a pair of arguments expressed by two different participants in a debate on a certain topic. For instance, given the argument "Making Internet a right only benefits society" (that we consider as H as a starting point), participants can be in favor of it (expressing arguments from which H can be inferred, as in Example 1), or can contradict such argument (expressing an opinion against it, as in Example 2). Since in debates one participant's argument comes after the other, we can extract such arguments and compare them both w.r.t. the main issue, and w.r.t. the other participants' arguments (when the new argument entails or contradicts one of the arguments previously expressed by another participant). For instance, given the same debate as before, a new argument T3 may be expressed by a third participant to contradict T2 (that becomes the new H (H1) in the pair), as shown in Example 4.

Example 4 (Continued)

T3*: I've seen the growing awareness within the developing world that computers and connectivity matter and can be useful. It's not that computers matter more than water, food, shelter and healthcare, but that the network and PCs can be used to ensure that those other things are available. Satellite imagery sent to a local computer can help villages find fresh water, mobile phones can tell farmers the prices at market so they know when to harvest.*
T2 $\equiv$ ***H1****: Internet not as important as real rights. We may think of such trivial things as a fundamental right, but consider the truly impoverished and what is most important to them. The right to vote, the right to liberty and freedom from slavery or the right to elementary education.*

With respect to the goal of our work, TE provides us with the techniques to identify the arguments in a debate, and to detect which kind of relation underlies each couple of arguments. A TE system returns indeed a judgment (entailment or contradiction) on the arguments pairs related to a certain topic, that are used as input to build the argumentation framework, as described in the next section. Example 5 presents how we combine TE with bipolar argumentation to compute at the end the set of accepted arguments.

Example 5 (Continued)

The textual entailment phase returns the following couples for the natural language opinions detailed in Examples 1, 2, and 4:

- *T1 entails H*
- *T2 attacks H*
- *T3 attacks H1 (i.e., T2)*

Given this result, the argumentation module of our framework maps each element to its corresponding argument: H $\equiv A_1$*, T1* $\equiv A_2$*, T2* $\equiv A_3$*, and T3* $\equiv A_4$*. The resulting argumentation framework, visualized in Figure 4.2, shows that the accepted arguments (using admissibility-based semantics) are* $\{A_1, A_2, A_4\}$*. This means that the issue "Making Internet a right only benefits society"' A_1 is considered as accepted. Double bordered arguments are the accepted ones.*

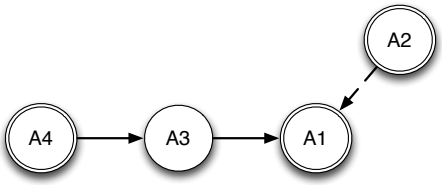


Figure 4.2: The argumentation framework built from the results of the TE module for Examples 1, 2, and 4.

4.1.4 Experimental setting

As a case study to experiment the combination of TE and argumentation theory to support the interaction of participants in online debates, we select Debatepedia, an encyclopedia of pro and con arguments on critical issues. In the following, we describe the creation of the data set of T-H pairs extracted from a sample of Debatepedia topics, the TE system we use, and we report on obtained results.

Data set. To create the data set of arguments pairs to evaluate our task, we follow the criteria defined and used by the organizers of RTE (see Section 4.1.1). To test the progress of TE systems in a comparable setting, the participants to RTE are provided with data sets composed of T-H pairs involving various levels of entailment reasoning (e.g., lexical, syntactic), and TE systems are required to produce a correct judgment on the given pairs (i.e. to say if the meaning of one text snippet can be inferred from the other). The data available for the RTE challenges are not suitable for our goal, since the pairs are extracted from news and are not linked among each others (i.e. they do not report opinions on a certain topic).

For this reason, we created a data set to evaluate our combined approach focusing on Debatepedia. We manually selected a set of topics (Table 4.21 column *Topics*) of Debatepedia debates, and for each topic we apply the following procedure:

1. the main issue (i.e., the title of the debate in its affirmative form) is considered as the starting argument;
2. each user opinion is extracted and considered as an argument;
3. since *attack* and *support* are binary relations, the arguments are coupled with:
   (a) the starting argument, or
   (b) other arguments in the same discussion to which the most recent argument refers (i.e., when a user opinion supports or attacks an argument previously expressed by another user, we couple the former with the latter), following the chronological order to maintain the dialogue structure;
4. the resulting pairs of arguments are then tagged with the appropriate relation, i.e., *attack* or *support*[5].

Using Debatepedia as case study provides us with already annotated arguments (*pro* $\Rightarrow$ *entailment*[6], and *con* $\Rightarrow$ *contradiction*), and casts our task as a yes/no entailment task. To show a step-by-step application of the procedure, let us consider the debated issue *Can coca be classified as a narcotic?*. At step 1, we transform its title into the affirmative form, and we consider it as the starting argument (a). Then, at step 2, we extract all the users opinions concerning this issue (both pro and con), e.g., (b), (c) and (d):

Example 6
(a) *Coca can be classified as a narcotic.*

(b) *In 1992 the World Health Organization's Expert Committee on Drug Dependence (ECDD) undertook a "prereview" of coca leaf at its 28th meeting. The 28th ECDD report concluded that, "the coca leaf is appropriately scheduled as a narcotic under the Single Convention on Narcotic Drugs, 1961, since cocaine is readily extractable from the leaf." This ease of extraction makes coca and cocaine inextricably linked. Therefore, because cocaine is defined as a narcotic, coca must also be defined in this way.*

(c) *Coca in its natural state is not a narcotic. What is absurd about the 1961 convention is that it considers the coca leaf in its natural, unaltered state to be a narcotic. The paste or the concentrate that is extracted from the coca leaf, commonly known as cocaine, is indeed a*

[5]The data set is freely available at `http://bit.ly/debatepedia_ds`.

[6]Here we consider only arguments implying another argument. Arguments "supporting" another argument, but not inferring it will be discussed in Section 4.1.5.

narcotic, but the plant itself is not.

(d) *Coca is not cocaine. Coca is distinct from cocaine. Coca is a natural leaf with very mild effects when chewed. Cocaine is a highly processed and concentrated drug using derivatives from coca, and therefore should not be considered as a narcotic.*

At step 3a we couple the arguments (b) and (d) with the starting issue since they are directly linked with it, and at step 3b we couple argument (c) with argument (b), and argument (d) with argument (c) since they follow one another in the discussion (i.e. user expressing argument (c) answers back to user expressing argument (b), so the arguments are concatenated - the same for arguments (d) and (c)).
At step 4, the resulting pairs of arguments are then tagged with the appropriate relation: **(b)** *supports* **(a)**, **(d)** *attacks* **(a)**, **(c)** *attacks* **(b)** and **(d)** *supports* **(c)**.

We collected 200 T-H pairs (Table 4.21), 100 to train and 100 to test the TE system (each data set is composed by 55 entailment and 45 contradiction pairs). The pairs considered for the test set concern completely new topics, never seen by the system.

Training set				
Topic	**#argum**	**#pairs**		
		TOT.	**yes**	**no**
Violent games boost aggressiveness	16	15	8	7
China one-child policy	11	10	6	4
Consider coca as a narcotic	15	14	7	7
Child beauty contests	12	11	7	4
Arming Libyan rebels	10	9	4	5
Random alcohol breath tests	8	7	4	3
Osama death photo	11	10	5	5
Privatizing social security	11	10	5	5
Internet access as a right	15	14	9	5
TOTAL	109	**100**	**55**	**45**
Test set				
Topic	**#argum**	**#pairs**		
		TOT.	**yes**	**no**
Ground zero mosque	9	8	3	5
Mandatory military service	11	10	3	7
No fly zone over Libya	11	10	6	4
Airport security profiling	9	8	4	4
Solar energy	16	15	11	4
Natural gas vehicles	12	11	5	6
Use of cell phones while driving	11	10	5	5
Marijuana legalization	17	16	10	6
Gay marriage as a right	7	6	4	2
Vegetarianism	7	6	4	2
TOTAL	110	**100**	**55**	**45**

Table 4.1: The Debatepedia data set used in our experiments.

TE system. To detect which kind of relation underlies each couple of arguments, we take advantage of the modular architecture of the EDITS system (Edit Distance Textual Entailment

		Train			Test		
	rel	*Pr.*	*Rec.*	*Acc.*	*Pr.*	*Rec.*	*Acc.*
EDITS	*yes*	0.71	0.73	**0.69**	0.69	0.72	**0.67**
	no	0.66	0.64		0.64	0.6	
WordOverl.	*yes*	0.64	0.65	0.61	0.64	0.67	0.62
	no	0.56	0.55		0.58	0.55	

Table 4.2: Systems performances on the Debatepedia data set (precision, recall and accuracy)

Suite) version 3.0, an open-source software package for recognizing TE[7] [255]. EDITS implements a distance-based framework which assumes that the probability of an entailment relation between a given T-H pair is inversely proportional to the distance between T and H (i.e., the higher the distance, the lower is the probability of entailment).[8] Within this framework the system implements different approaches to distance computation, i.e., both edit distance algorithms (that calculate the T-H distance as the cost of the edit operations, i.e., insertion, deletion and substitution that are necessary to transform T into H), and similarity algorithms. Each algorithm returns a normalized distance score. At a training stage, distance scores calculated over annotated T-H pairs are used to estimate a threshold that best separates positive from negative examples. Such threshold is then used at a test stage to assign a judgment and a confidence score to each test pair.

Evaluation. To evaluate our combined approach, we carry out a two-step evaluation: first, we assess the performances of the TE system to correctly assign the entailment and contradiction relations to the pairs of arguments in the Debatepedia data set. Then, we evaluate how much such performances impact on the application of the argumentation theory module, i.e. how much a wrong assignment of a relation to a pair of arguments is propagated in the argumentation framework.

For the first evaluation, we run EDITS on the Debatepedia training set to learn the model, and we test it on the test set. We tuned EDITS in the following configuration: *i)* cosine similarity as the core distance algorithm, *ii)* distance calculated on lemmas, and *iii)* a stopword list is defined to set no distance between stopwords. We use the system off-the-shelf, applying one of its basic configurations. As future work, we plan to fully exploit EDITS features, integrating background and linguistic knowledge in the form of entailment rules, and to calculate the distance between T and H on their syntactic structure.

Table 4.2 reports on the obtained results both using EDITS and using a baseline that applies a Word Overlap algorithm on tokenized text. Even using a basic configuration of EDITS, and a small data set (100 pairs for training) performances on Debatepedia test set are promising, and in line with performances of TE systems on RTE data sets (usually containing about 1000 pairs for training and 1000 for test). In order to understand if increasing the number of argument pairs in the training set could bring to an improvement in the system performances, the EDITS learning curve is visualized in Figure 4.3. Note that augmenting the number of training pairs actually improves EDITS accuracy on the test set, meaning that we should consider extending the Debatepedia data set for future work.

As a second step in our evaluation phase, we consider the impact of EDITS performances on the acceptability of the arguments, i.e. how much a wrong assignment of a relation to a

[7] `http://edits.fbk.eu/`

[8] In previous RTE challenges, EDITS always ranked among the 5 best participating systems out of an average of 25 systems, and is one of the few RTE systems available as open source `http://aclweb.org/aclwiki/index.php?title=Textual_Entailment_Resource_Pool`

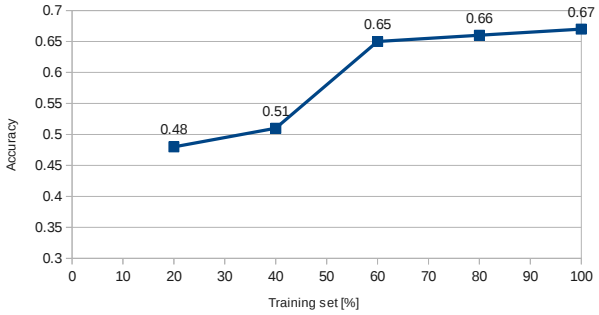


Figure 4.3: EDITS learning curve on Debatepedia data set

pair of arguments affects the acceptability of the arguments in the argumentation framework. We use admissibility-based semantics to identify the accepted arguments both on the correct argumentation framework of each Debatepedia topic (where entailment/contradiction relations are correctly assigned, i.e. the goldstandard), and on the framework generated assigning the relations resulted from the TE system judgments. The precision of the combined approach we propose in the identification of the accepted arguments is on average 0.74 (i.e. arguments accepted by the combined system and by the goldstandard w.r.t. a certain Debatepedia topic), and the recall is 0.76 (i.e. arguments accepted in the goldstandard and retrieved as accepted by the combined system). Its accuracy (i.e. ability of the combined system to accept some arguments and discard some others) is 0.75, meaning that the TE system mistakes in relation assignment propagate in the argumentation framework, but results are still satisfying.

4.1.5 Extending the analysis on bipolar argumentation beyond TE

In the previous section, we assumed the TE relation extracted from NL texts as equivalent to the support relation in bipolar argumentation. On closer view, this is a strong assumption. In this second part of our work, we aim at verifying on an extended sample of real data from Debatepedia whether it is always the case that support is equivalent to TE. In particular, for addressing this issue, we focus both on the relations between support and entailment, and on the relations between attack and contradiction. We extend the data set presented before, extracting an additional set of arguments from Debatepedia topics. Even if our data set cannot be exhaustive, the methodology we apply for the arguments extraction aims at preserving the original structure of the debate, to make it as representative as possible of daily human interactions in natural language.

Two different empirical studies are presented in this section. The first one follows the analysis presented in Section 4.1.3, and explores the relation among the notion of *support* and *attack* in bipolar argumentation, and the *semantic inferences* as defined in NLP. The second analysis starts instead from the comparative study of [102] of the four complex attacks proposed in the literature (see Section 4.1.2), and investigates their distribution in NL debates.

Data set. We select the same topics as in Section 4.1.4, since this is the only freely available data set of natural language arguments (Table 4.21, column *Topics*). But Since that data set was created respecting the assumption that the TE relation and the support relation are equivalent, in all the previously collected pairs both TE and support relations (or contradiction and attack relations) hold.

In this study we want to move a step further, to understand whether it is always the case that support is equivalent to TE (and contradiction to attack). We therefore apply again the extraction methodology described in Section 4.1.4 to extend our data set. In total, our new data set contains 310 different arguments and 320 argument pairs (179 expressing the *support*

relation among the involved arguments, and 141 expressing the *attack* relation, see Table 4.21). We consider the obtained data set as representative of human debates in a non-controlled setting (Debatepedia users position their arguments with respect to the others as PRO or CON, the data are not biased), and we use it for our empirical studies.

Debatepedia data set		
Topic	**#argum**	**#pairs**
Violent games boost aggressiveness	17	23
China one-child policy	11	14
Consider coca as a narcotic	17	22
Child beauty contests	13	17
Arming Libyan rebels	13	15
Random alcohol breath tests	11	14
Osama death photo	22	24
Privatizing social security	12	13
Internet access as a right	15	17
Ground zero mosque	11	12
Mandatory military service	15	17
No fly zone over Libya	18	19
Airport security profiling	12	13
Solar energy	18	19
Natural gas vehicles	16	17
Use of cell phones while driving	16	16
Marijuana legalization	23	25
Gay marriage as a right	10	10
Vegetarianism	14	13
TOTAL	310	**320**

Table 4.3: Debatepedia data set.

First study: support and TE. Our first empirical study aims at a better understanding of the relation among the notion of support in bipolar argumentation [102], and the definition of semantic inference in NLP (in particular, the more specific notion of TE) [132].

Basing on the TE definition, an annotator with skills in linguistics has carried out a first phase of annotation of the Debatepedia data set. The goal of such annotation is to individually consider each pair of *support* and *attack* among arguments, and to additionally tag them as *entailment*, *contradiction* or *null*. The *null* judgment can be assigned in case an argument is supporting another argument without inferring it, or the argument is attacking another argument without contradicting it. As exemplified in Example 6, a correct entailment pair is **(b)** $\Rightarrow$ **(a)**, while a contradiction is **(d)** $\nRightarrow$ **(a)**. A *null* judgment is assigned to **(d) - (c)**, since the former argument supports the latter without inferring it. Our data set is an extended version of [90]'s one allowing for a deeper investigation.

To assess the validity of the annotation task, we calculate the inter-annotator agreement. Another annotator with skills in linguistics has therefore independently annotated a sample of 100 pairs of the data set. We calculated the inter-annotator agreement considering the argument pairs tagged as *support* and *attacks* by both annotators, and we verify the agreement between the pairs tagged as *entailment* and as *null* (i.e. no entailment), and as *contradiction* and as *null* (i.e. no contradiction), respectively. Applying κ to our data, the agreement for our task is $\kappa = 0.74$. As a rule of thumb, this is a satisfactory agreement.

Table 4.4 reports the results of the annotation on our Debatepedia data set, as resulting after a reconciliation phase carried out by the annotators[9].

[9]In this phase, the annotators discuss the results to find an agreement on the annotation to be released.

Relations		% arguments (# arg.)
support	*+ entailment*	61.6 (111)
	- entailment (null)	38.4 (69)
attack	*+ contradiction*	71.4 (100)
	- contradiction (null)	28.6 (40)

Table 4.4: Support and TE relations on Debatepedia data set.

On the 320 pairs of the data set, 180 represent a *support* relation, while 140 are *attacks*. Considering only the *supports*, we can see that 111 argument pairs (i.e., 61.6%) are an actual entailment, while in 38.4% of the cases the first argument of the pair supports the second one without inferring it (e.g., **(d) - (c)** in Example 6). With respect to the *attacks*, we can notice that 100 argument pairs (i.e., 71.4%) are both attack and contradiction, while only the 28.6% of the argument pairs does not contradict the arguments they are attacking, as in Example 7.

Example 7
(e) *Coca chewing is bad for human health. The decision to ban coca chewing fifty years ago was based on a 1950 report elaborated by the UN Commission of Inquiry on the Coca Leaf with a mandate from ECOSOC: "We believe that the daily, inveterate use of coca leaves by chewing is thoroughly noxious and therefore detrimental".*
(f) *Chewing coca offers an energy boost. Coca provides an energy boost for working or for combating fatigue and cold.*

Differently from the relation between support-entailment, the difference between attack and contradiction is more subtle, and it is not always straightforward to say whether an argument attacks another argument without contradicting it. In Example 7, we consider that **(e)** does not contradict **(f)** even if it attacks **(f)**, since chewing coca can offer an energy boost, and still be bad for human health. This kind of attacks is less frequent than the attacks-contradictions (see Table 4.4).

Considering the three way scenario to map TE relation with bipolar argumentation, argument pairs connected by a relation of support (but where the first argument does not entail the second one), and argument pairs connected by a relation of attack (but where the first argument does not contradict the second one) have to be mapped as *unknown* pairs in the TE framework. The *unknown* relation in TE refers to the T-H pairs where the entailment cannot be determined because the truth of H cannot be verified on the basis of the content of T. This is a broad definition, that can apply also to pairs of non related sentences (that are considered as unrelated arguments in bipolar argumentation).

From an application viewpoint, as highlighted in [390] and [222], argumentation theory should be used as a tool in on-line discussions applications to identify the relations among the statements, and provide a structure to the dialogue to easily evaluate the user's opinions. Starting from the methodology proposed in Section 4.1.3 for passing from natural language arguments to a bipolar argumentation framework, our study demonstrates that applying the TE approach would be productive in the 66% of the Debatepedia data set. Other techniques should then be experimented to cover the other cases, for instance measuring the semantic relatedness of the two propositions using Latent Semantics Analysis techniques [263].

Second study: complex attacks. We carry out now a comparative evaluation of the four additional attacks proposed in the literature, and we investigate their meaning and distribution on the sample of NL arguments.

Basing on the additional attacks (Section 4.1.2), and the original AF of each topic in our data set (Table 4.21), the following procedure is applied: the *supported* (secondary, mediated, and extended, respectively) attacks are added, and the argument pairs resulting from coupling the arguments linked by this relation are collected in the data set "supported (secondary, mediated, and extended, respectively) attack". Collecting the argument pairs generated from the different types of complex attacks in separate data sets allows us to independently analyze each type, and to perform a more accurate evaluation.[10] Figures 4.4a-d show the four AFs resulting from the addition of the complex attacks in the example *Can coca be classified as a narcotic?*. Note that the AF in Figure 4.4a, where the supported attack is introduced, is the same of Figure 4.4b where the mediated attack is introduced. Notice that, even if the additional attack which is introduced coincide, i.e., d attacks b, this is due indeed to different interactions among supports and attacks (as highlighted in the figure), i.e., in the case of supported attacks this is due to the support from d to c and the attack from c to b, while in the case of mediated attacks this is due to the support from b to a and the attack from d to a.

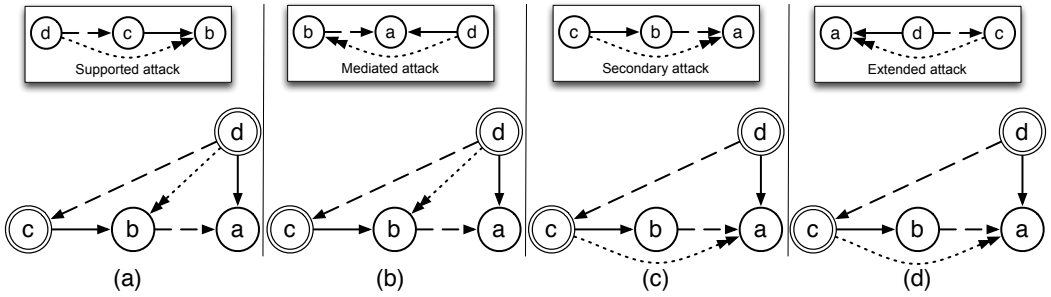


Figure 4.4: The bipolar argumentation framework with the introduction of complex attacks. The top figures show which combination of support and attack generates the new additional attack.

A second annotation phase is then carried out on the data set, to verify if the generated argument pairs of the four data sets are actually attacks (i.e., if the models of complex attacks proposed in the literature are represented in real data). More specifically, an argument pair resulting from the application of a complex attack can be annotated as: *attack* (if it is a correct attack) or as *unrelated* (in case the meanings of the two arguments are not in conflict). For instance, the argument pair **(g)-(h)** (Example 8) resulting from the insertion of a *supported* attack, cannot be considered as an attack since the arguments are considering two different aspects of the issue.

Example 8 ***(g)*** *Chewing coca offers an energy boost. Coca provides an energy boost for working or for combating fatigue and cold.*
(h) *Coca can be classified as a narcotic.*

In the annotation, *attacks* are then annotated also as *contradiction* (if the first argument contradicts the other) or *null* (in case the first argument does not contradict the argument it is attacking, as in Example 7). Due to the complexity of the annotation, the same annotation task has been independently carried out also by a second annotator, so as to compute inter-annotator agreement. It has been calculated on a sample of 80 argument pairs (20 pairs randomly extracted from each of the "complex attacks" data set), and it has the goal to assess the validity of the annotation task (counting when the judges agree on the same annotation).

[10]Data sets freely available for research purposes at `http://bit.ly/VZIs6M`

We calculated the inter-annotator agreement for our annotation task in two steps. We *(i)* verify the agreement of the two judges on the argument pairs classification *attacks*/*unrelated*, and *(ii)* consider only the argument pairs tagged as *attacks* by both annotators, and we verify the agreement between the pairs tagged as *contradiction* and as *null* (i.e. no contradiction). Applying κ to our data, the agreement for the first step is $\kappa = 0.77$, while for the second step $\kappa = 0.71$. As a rule of thumb, both agreements are satisfactory, although they reflect the higher complexity of the second annotation (*contradiction*/*null*), as pointed out in Section 4.1.5.

The distribution of complex attacks in the Debatepedia data set, as resulting after a reconciliation phase carried out by the annotators, is shown in Table 4.5. As can be noticed, the *mediated* attack is the most frequent type of attack, generating 335 new argument pairs in the NL sample we considered (i.e. the conditions that allow the application of this kind of complex attacks appear more frequently in real debates). Together with *secondary* attacks, they appear in the AFs of all the debated topics. On the contrary, *extended* attacks are added in 11 out of 19 topics, and *supported* attacks in 17 out of 19 topics. Considering all the topics, on average only 6 pairs generated from the additional attacks were already present in the original data set, meaning that considering also these attacks is a way to hugely enrich our data set of NL debates.

Proposed models	**# occ.**	**attacks**		**unrelated**
		+ contr (null)	*- contr (null)*	
Supported attacks	47	23	17	7
Secondary attacks	53	29	18	6
Mediated attacks	335	84	148	103
Extended attacks	28	15	10	3

Table 4.5: Complex attacks distribution in our data set.

Figure 4.5 graphically represents the complex attacks distribution. Considering the first step of the annotation (i.e. *attacks* vs *unrelated*), the figure shows that the latter case is very infrequent, and that (except for *mediated* attacks) on average only 10% of the argument pairs are tagged as *unrelated*. This observation can be considered as a proof of concept of the four theoretical models of complex attacks we analyzed. Due to the fact that the conditions for the application of the *mediated* attacks are verified more often in the data, it has the drawback of generating more unrelated pairs. Still, the number of successful cases is high enough to consider this kind of attack as representative of human interactions. Considering the second step of the annotation (i.e. *attacks* as *contradiction* or *null*), we can see that results are in line with those reported in our first study (Table 4.4), meaning that also among complex attacks the same distribution is maintained.

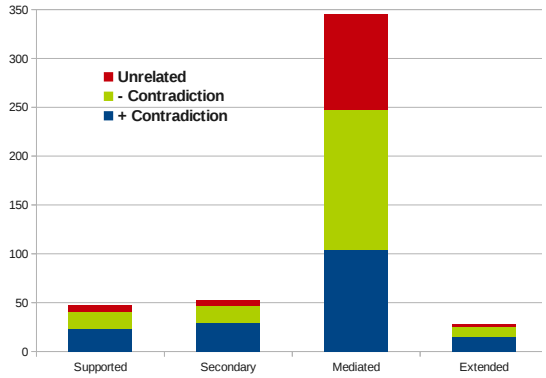


Figure 4.5: Complex attacks distribution in our data set.

4.2 Argument Mining on Twitter: arguments, facts and sources

Social media platforms like Twitter[11] and newspapers blogs allow users to post their own viewpoints on a certain topic, or to disseminate news read on newspapers. Being these texts short, without standard spelling and with specific conventions (e.g., hashtags, emoticons), they represent an open challenge for standard argument mining approaches [414]. The nature and peculiarity of social media data rise also the need of defining new tasks in the argument mining domain [1, 285].

In this work, we tackle the first standard task in argument mining, addressing the research question: *how to mine arguments from Twitter?* Going a step further, we address also the following sub-questions that arise in the context of social media: *i)* how to distinguish factual arguments from opinions? *ii)* how to automatically detect the source of factual arguments? To answer these questions, we extend and annotate a dataset of tweets extracted from the streams about the Grexit and the Brexit news. To address the first task of argument detection, we apply supervised classification to separate *argument-tweets* from non-argumentative ones. By considering only argument-tweets, in the second step we apply again a supervised classifier to recognize tweets reporting factual information from those containing opinions only. Finally, we detect, for all those arguments recognized as factual in the previous step, what is the source of such information (e.g., the CNN), relying on the type of the Named Entities recognized in the tweets. The last two steps represent new tasks in the argument mining research field, of particular importance in social media applications.

In the following, we describe the approaches we have developed to address the following tasks: *i)* Argument detection, *ii)* Factual vs opinion classification, and *iii)* Source identification, on social media data. Our experimental setting - whose goal is to investigate the tasks' feasibility on such peculiar data - considers a dataset of tweets related to the political debates on whether or not Great Britain and Greece had to leave the European Union (i.e. #Brexit and #Grexit threads in Twitter).

4.2.1 Experimental setting

Dataset. The only available resource of annotated tweets for argument mining is DART [68]. From the highly heterogeneous topics contained in such resource (i.e. the letter to Iran written by 47 U.S. senators; the referendum for or against Greece leaving the EU; the release of Apple iWatch; the airing of the 4th episode of the 5th season of the TV series Game of Thrones), and considering the fact that tweets discussing a political topic generally have a more developed argumentative structure than tweets commenting on a product release, we decided to select for our experiments the subset of the DART dataset on the thread *#Grexit* (987 tweets). Then, following the same methodology described in [68], we have extended such dataset collecting 900 tweets from the thread on *#Brexit*. From the original thread, we filtered away retweets, accounts with a bot probability >0.5 [136], and almost identical tweets (Jaccard distance, empirically evaluated threshold). Given that tweets in DART are already annotated for task 1 (argument/non-argument, see Section 4.2.1), two annotators carried out the same task on the newly extracted data. Moreover, the same annotators annotated both datasets (Grexit/Brexit) for the other two tasks of our experiments, i.e. *i)* given the argument tweets, annotation of tweets as either containing factual information or opinions, and *ii)* given factual argument tweets, annotate their source when explicitly cited. Tables 4.6, 4.7 and 4.8 contain statistical information on the datasets.

[11] `www.twitter.com`

Inter annotator agreement (IAA) [98] between the two annotators has been calculated for the three annotation tasks, resulting in κ=0.767 on the first task (calculated on 100 tweets), κ=0.727 on the second task (on 80 tweets), and Dice=0.84 [145][12] on the third task (on the whole dataset). More specifically, to compute IAA, we sampled the data applying the same strategy: for the first task, we randomly selected 10% of the tweets of the Grexit dataset (our training set); for task 2, again we randomly selected 10% of the tweets annotated as argument in the previous annotation step; for task 3, given the small size of the dataset, both annotators annotated the whole corpus.

dataset	# argument	# non-arg	total
Brexit	713	187	900
Grexit	746	241	987
total	1459	428	1887

Table 4.6: Dataset for task 1: argument detection

dataset	# factual arg.	# opinion	total
Brexit	138	575	713
Grexit	230	516	746
total	368	1091	1459

Table 4.7: Dataset for task 2: factual arguments vs opinions classification

dataset	# arg. with source cit.	# arg. without source cit.	total
Brexit	40	98	138
Grexit	79	151	230
total	119	249	368

Table 4.8: Dataset for task 3: source identification

Classification algorithms. We tested Logistic Regression (LR) and Random Forest (RF) classification algorithms, relying on the *scikit-learn* tool suite[13]. For the learning methods, we have used a Grid Search (exhaustive) through a set of predefined hyper-parameters to find the best performing ones (the goal of our work is not to optimize the classification performance but to provide a preliminary investigation on new tasks in argument mining over Twitter data). We extract argument-level features from the dataset of tweets (following [466]), that we group into the following categories:

- *Lexical (L):* unigram, bigram, WordNet verb synsets;
- *Twitter-specific (T):* punctuation, emoticons;

[12]Dice is used instead of κ to account for partial agreement on the set of sources detected in the tweets.
[13]`http://scikit-learn.org/`

- *Syntactic/Semantic (S):* we have two versions of dependency relations as features, one being the original form, the other generalizing a word to its POS tag in turn. We also use the syntactic tree of the tweets as feature. We apply the Stanford parser [298] to obtain parse trees and dependency relations;

- *Sentiment (SE):* we extract the sentiment from the tweets with the Alchemy API[14], the sentiment analysis feature of IBM's Semantic Text Analysis API. It returns a polarity label (positive, negative or neutral) and a polarity score between -1 (totally negative) and 1 (totally positive).

As baselines we consider both LR and RF algorithms with a set of basic features (i.e., lexical).

Task 1: Argument detection. The task consists in classifying a tweet as being an argument or not. We consider as arguments all those text snippets providing a portion of a standard argument structure, i.e., opinions under the form of claims, facts mirroring the data in the Toulmin model of argument [442], or persuasive claims, following the definition of argument tweet provided in [68, 69]. Our dataset contains 746 argument tweets and 241 non-argument tweets for Grexit (that we use as training set), and 713 argument tweets and 187 non-argument tweets for Brexit (the test set). Below we report an example of argument tweet (a), and of a non-argument tweet (b).

(a) *Junker asks "who does he think I am". I suspect elected PM Tsipras thinks Junker is an unelected Eurocrat. #justsaying #democracy #grexit*
(b) *#USAvJPN #independenceday #JustinBieberBestIdol Macri #ConEsteFrioYo happy 4th of july #Grefenderum Wireless Festival*

We cast the argument detection task as a binary classification task, and we apply the supervised algorithms described in Section 4.2.1. Table 4.9 reports on the obtained results with the different configurations, while Table 4.10 reports on the results obtained by the best configuration, i.e., LR + All features, per each category.

Approach	Precision	Recall	F1
RF+L	0.76	0.69	0.71
LR+L	0.76	0.71	0.73
LR+all features	0.80	0.77	**0.78**

Table 4.9: Results obtained on the test set for the argument detection task (L=lexical features)

Category	P	R	F1	#arguments per category
non-arg	0.46	0.60	0.52	187
arg	0.89	0.82	0.85	713
avg/total	0.80	0.77	**0.78**	900

Table 4.10: Results obtained by the best model on each category of the test set for the argument detection task

Most of the miss-classified tweets are either ironical, e.g.:

[14] `https://www.ibm.com/watson/alchemy-api.html`

If #Greece had a euro for every time someone mentioned #Grexit and #Greferendum they would probably have enough for a bailout. #GreekCrisis

that was wrongly classified as argument, or contain reported speech, e.g.:

Jeremy Warner: Unintentionally, the Greeks have done themselves a favour. Soon, they will be out of the euro `http://t.co/YmqXi36lGj` *#Grexit*

that was wrongly classified as non argument. Our results are comparable to those reported in [69] (they trained a supervised classifier on the tweets of all topics in the DART dataset but the iWatch, used as test set). Better performances obtained in our setting are most likely due to a better feature selection, and to the fact that in our case the topics in the training and test sets are more homogeneous.

Task 2: Factual vs opinion classification This task consists in classifying argument-tweets as containing factual information or being opinion-based [363]. Our interest focuses in particular on factual argument-tweets, as we are interested then in the automated identification of their sources. This would allow then to rank factual tweet-arguments depending on the reliability or expertise of their source for subsequent tasks as fact checking. Given the huge amount of work in the literature devoted to opinion extraction, we do not address any further analysis on opinion-based arguments here, referring the interested reader to [281].

An argument is annotated as *factual* if it contains a piece of information which can be proved to be true (see example (a) below), or if it contains "reported speech" (see example (b) below). All the other argument tweets are considered as "opinion" (see example (c) below).

(a) *72% of people who identified as "English" supported #Brexit (while no majority among those identifying as "British")* `https://t.co/MuUXqncUBe`
(b) *#Hollande urges #UK to start #Brexit talks as soon as possible.* `https://t.co/d12TV8JqYD`.
(c) *Trump is going to sell us back to England. #Brexit #RNCinCLE*

Our dataset contains 230 factual argument tweets and 516 opinion argument tweets for Grexit (training set), and 138 factual argument tweets and 575 opinion argument tweets for Brexit (test set).

To address the task of factual vs opinion arguments classification, we apply the supervised classification algorithms previously described. Tweets from Grexit dataset are used as training set, and those from Brexit dataset as test set. Table 4.11 reports on the obtained results, while Table 4.12 reports on the results obtained by the best configuration, i.e. LR + All features, per each category.

Approach	Precision	Recall	F1
RF+L	0.75	0.68	0.71
LR+L	0.75	0.75	0.75
LR+all features	0.81	0.79	**0.80**

Table 4.11: Results obtained on the test set for the factual vs opinion argument classification task (L=lexical features)

Most of the miss-classified tweets contain reported opinions/reported speech and are wrongly classified by the algorithm as opinion - such behaviour could be expected given that sentiment features play a major role in these cases, e.g.,

Category	P	R	F1	#arguments per category
fact	0.49	0.50	0.50	138
opinion	0.88	0.87	0.88	575
avg/total	0.81	0.79	**0.80**	713

Table 4.12: Results obtained by the best model on each category of the test set for the factual vs opinion argument classification task

Thomas Piketty accuses Germany of forgetting history as it lectures Greece `http://t.co/B0UqPn0i6T` *#grexit*

Again, the other main reason for miss-classification is sarcasm/irony contained in the tweets, e.g.,

So for Tsipras, no vote means back to the table, for Varoufakis, meant Grexit?

that was wrongly classified as fact.

Task 3: Source identification Since factual arguments (as defined above) are generally reported by news agencies and individuals, the third task we address - and that can be of a value in the context of social media - is the recognition of the information source that disseminates the news reported in a tweet (when explicitly mentioned). For instance, in:

The Guardian: Greek crisis: European leaders scramble for response to referendum no vote. `http://t.co/cUNiyLGfg3`

the source of information is The Guardian newspaper. Such annotation is useful to rank factual tweet-arguments depending on the reliability or expertise of their source in news summarization or fact-checking applications, for example.

Our dataset contains 79 factual argument tweets where the source is explicitly cited for Grexit (training set), and 40 factual argument tweets where the source is explicitly cited for Brexit (test set). Given the small size of the available annotated dataset, to address this task we implemented a simple string matching algorithm that relies on a gazetteer containing a set of Twitter usernames and hashtags extracted from the training data, and a list of very common news agencies (e.g., BBC, CNN, CNBC). If no matches are found, the algorithm extracts the NEs from the tweets through [346]'s system, and applies the following two heuristics: *i)* if a NE is of type `dbo:Organisation` or `dbo:Person`, it considers such NE as the source; *ii)* it searches in the abstract of the DBpedia[15] page linked to that NE if the words "news", "newspaper" or "magazine" appear (if found, such entity is considered as the source). In the example above, the following NEs have been detected in the tweet: "The Guardian" (linked to the DBpedia resource `http://dbpedia.org/page/The\_Guardian`) and "Greek crisis" (linked to `http://dbpedia.org/page/Greek\_government-debt\_crisis`). Applying the mentioned heuristics, the first NE is considered as the source. Table 4.13 reports on the obtained results. As baseline, we use a method that considers all the NEs detected in the tweet as sources.

Most of the errors of the algorithm are due to information sources not recognized as NEs (in particular, when the source is a Twitter user), or NEs that are linked to the wrong DBpedia

[15] `http://www.dbpedia.org`

Approach	Precision	Recall	F1
Baseline	0.26	0.48	0.33
Matching+heurist.	0.69	0.64	**0.67**

Table 4.13: Results obtained on the test set for the source identification task

page. However, in order to draw more interesting conclusions on the most suitable methods to address this task, we would need the increase the size of the dataset.

4.3 Argumentation analysis for political speeches

While some of the above mentioned AM approaches have been proposed to detect claims in political debates, e.g., [278, 338], little attention has been devoted to the prediction of relations between arguments, which could help historians, social and political scientists in the analysis of argumentative dynamics (e.g., supports, attacks) between parties and political opponents. For example, this analysis could support the study of past political speeches and of the repercussions of such claims over time. It could also be used to establish relations with the current way of debating in politics. In order to find argumentation patterns in political speeches, typically covering a wide range of issues from international politics to environmental challenges, the application of computational methods to assist scholars in their qualitative analysis is advisable. In this work, we tackle the following research question: *To what extent can we apply argument mining models to support and ease the analysis and modeling of past political speeches?* This research question breaks down into the following subquestions:

- Given a transcription of speeches from different politicians on a certain topic, how can we automatically predict the relation holding between two arguments, even if they belong to different speeches?
- How can the output of the above-mentioned automated task be used to support history and political science scholars in the curation, analysis and editing of such corpora?

This issue is investigated by creating and analysing a new annotated corpus for this task, based on the transcription of discourses and official declarations issued by Richard Nixon and John F. Kennedy during the 1960 US Presidential campaign. Moreover, we develop a relation classification system with specific features able to *predict support* and *attack* relations between arguments [279], distinguishing them from unrelated ones. This argumentation mining pipeline ends with the visualization of the resulting graph of the debated topic using the OVA$^+$ tool.[16]

The main contributions of this article are *(1)* an annotated corpus consisting of 1,462 pairs of arguments in natural language (around 550,000 tokens) covering 5 topics, *(2)* a feature-rich Support Vector Machines (SVM) model for relation prediction, and *(3)* an end-to-end workflow to analyse arguments that, starting from one or more monological corpora in raw text, outputs the argumentation graph of user-defined topics.

4.3.1 Corpus extraction and annotation

Since no data for this task were available, we collect the transcription of speeches and official declarations issued by Nixon and Kennedy during 1960 Presidential campaign from The American Presidency Project.[17] The corpus includes 881 documents, released under the NARA

[16] `http://ova.arg-tech.org/`

[17] The American Presidency Project (`http://www.presidency.ucsb.edu/1960_election.php`)

public domain license, and more than 1,6 million tokens (around 830,000 tokens for Nixon and 815,000 tokens for Kennedy). We select this document collection because of its relevance from a historical perspective: the 1960 electoral campaign has been widely studied by historians and political scientists, being the first campaign broadcast on television. The issues raised during the campaign shaped the political scenario of the next decades, for example the rising Cold War tensions between the United States and the Soviet Union or the relationship with Cuba.

Dataset creation. In order to include relevant topics in the dataset, we asked a history scholar to list a number of issues that were debated during 1960 campaign, around which argumentation pairs could emerge. With his help, we selected the following ones: *Cuba*, *disarmament*, *healthcare*, *minimum wage* and *unemployment* (henceforth *topics*). We then extracted pairs of candidate arguments as follows. For each topic, we manually define a set of keywords (e.g., *[medical care, health care]*) that lexically express the topic. Then, we extract from the corpus all sentences containing at least one of these keywords, plus the sentence before and after them to provide some context: each candidate argument consists then of a snippet of text containing three consecutive sentences and a date, corresponding to the day in which the original speech was given during the campaign.

In the following step, we combine the extracted snippets into pairs using two different approaches. Indeed, we want to analyse two different types of argumentations: those *between candidates*, and those emerging from the speeches uttered by the *same candidate* over time. In the first case, for each topic, we sort all the candidate arguments in chronological order, and then create pairs by taking one or more snippets by a politician and the one(s) immediately preceding it by his opponent. These data are thus shaped as a sort of indirect dialogue, in which Nixon and Kennedy talk about the same topics in chronological order. However, the arguments of a speaker are not necessarily the direct answer to the arguments of the other one, making it challenging to label the relation holding between the two.

In the second case, we sort by topic all the candidate arguments in chronological order, as in the previous approach. However, each candidate argument is paired with what the same politician said on the same topic in the immediately preceding date. These data provide information about how the ideas of Nixon and Kennedy evolve during the electoral campaign, showing, if any, shifts in their opinions. We follow these two approaches also with the goal to obtain a possibly balanced dataset: we expect to have more attack relations holding between pairs of arguments from different candidates, while pairs of arguments from the same candidate should be coherent, mainly supporting each other.

Through this pairing process, we obtain 4,229 pairs for the *Cuba* topic, 2,508 pairs for *disarmament*, 3,945 pairs for *health-care*, 6,341 pairs for *minimum wage*, and 2,865 pairs for *unemployment*, for a total of 19,888 pairs.

Annotation. From the pool of automatically extracted pairs, we manually annotate a subset of 1,907 pairs randomly selected over the five topics. Annotators were asked to mark if between two given arguments there was a relation of *attack* (see Example 9 on minimum wage), a relation of *support* (see Example 10 on disarmament) or if there was *no relation* (arguments are neither supporting, nor attacking each other, tackling different issues of the same topic).

Example 9

***Nixon**: And here you get the basic economic principles. If you raise the minimum wage, in my opinion - and all the experts confirm this that I have talked to in the Government - above $1.15, it would mean unemployment; unemployment, because there are many industries that could not pay more than $1.15 without cutting down their work force. $1.15 can be absorbed, and then at a later time we could move to $1.25 as the economy moves up.*

***Kennedy**: The fact of the matter is that Mr. Nixon leads a party which has opposed progress for 25 years, and he is a representative of it. He leads a party which in 1935 voted 90 percent against a 25-cent minimum wage. He leads a party which voted 90 percent in 1960 against $1.25 an hour minimum wage.*

Example 10
***Nixon**: I want to explain that in terms of examples today because it seems to me there has been a great lack of understanding in recent months, and, for that matter in recent years, as to why the United States has followed the line that it has diplomatically. People have often spoken to me and they have said, Why can't we be more flexible in our dealings on disarmament? Why can't we find a bold new program in this area which will make it possible for the Soviet Union to agree? The answer is that the reason the Soviet Union has not agreed is that they do not want apparently to disarm unless we give up the right to inspection.*
***Nixon**: People say, Now, why is it we can't get some imaginative disarmament proposals, or suspension of nuclear test proposals? Aren't we being too rigid? And I can only say I have seen these proposals over the years, and the United States could not have been more tolerant. We have not only gone an extra mile - we have gone an extra 5 miles - on the tests, on disarmament, but on everything else, but every time we come to a blocking point, the blocking point is no inspection, no inspection.*

The annotation guidelines included few basic instructions: if the statements cover more than one topic, annotators were asked to focus only on the text segments dealing with the chosen topic. Annotation was carried out by strictly relying on the content of the statements, avoiding personal interpretation. Examples of *attack* are pairs where the candidates propose two different approaches to reach the same goal, where they express different considerations on the current situation with respect to a problem, or where they have a different attitude with respect to the work done in the past. For example, in order to increase minimum wage, Nixon proposed to set it to 1.10$ per hour, while Kennedy opposed this initiative, claiming that 1.35$ should be the minimum wage amount. In this example, the opponents have the same goal, i.e., increase minimum wage, but their statements are annotated as an attack because their initiatives are different, clearly expressing their disagreement.

After an initial training following the above guidelines, 3 annotators were asked to judge a common subset of 100 pairs to evaluate inter-annotator agreement. This was found to be 0.63 (Fleiss' Kappa), which as a rule of thumb is considered a substantial agreement [264]. After that, each annotator judged a different set of argument pairs, with a total of 1,907 judgements collected. In order to balance the data, we discarded part of the pairs annotated with *no relation* (randomly picked).

Overall, the final annotated corpus[18] is composed of 1,462 pairs: 378 pairs annotated with *attack*, 353 pairs annotated with *support*, and 731 pairs where these relations do not hold. An overview of the annotated corpus is presented in Table 4.14.

Topic	Attack	Support	No Relation
Cuba	38	40	180
Disarmament	76	108	132
Medical care	75	72	142
Minimum wage	125	80	107
Unemployment	64	53	170

Table 4.14: Topic and class distribution in the annotated corpus

[18]The dataset is available at `https://dh.fbk.eu/resources/political-argumentation`

4.3.2 Experiments on relation prediction

To facilitate the construction of argument graphs and support the argumentative analysis of political speeches, we propose an approach to automatically label pairs of arguments according to the relation existing between them, namely *support* and *attack*.

Given the strategy adopted to create the pairs, the paired arguments may happen to be also unrelated (50% of the pairs are labeled with *no relation*). Therefore, we first isolate the pairs connected through a relation, and then we classify them as *support* or *attack*. Each step is performed by a binary classifier using specific features, which we describe in the following subsection. In this work, we present the results obtained with the feature set that achieved the best performance on 10-fold cross validation.

Experimental setting. The *first step* concerns the binary classification of related and unrelated pairs. In this step the pairs annotated with support and attack have been merged under the *related* label. We first pre-process all the pairs using the Stanford CoreNLP suite [298] for tokenization, lemmatization and part-of-speech tagging. Then, for each pair we define three sets of features, representing the lexical overlap between snippets, the position of the topic mention in the snippet, as a proxy for its relevance, and the similarity of snippets with other related / unrelated pairs.

Lexical overlap: the rationale behind this information is that two related arguments are supposed to be more lexically similar than unrelated ones. Therefore, we compute *i)* the number of nouns, verbs and adjectives shared by two snippets in a pair, normalized by their length, and *ii)* the normalized number of nouns, verbs and adjectives shared by the argument subtrees where the topic is mentioned.

Topic position: the rationale behind this information is that, if the same topic is central in both candidate arguments, then it is likely that these arguments are related. To measure this, we represent with a set of features how often the topic (expressed by a list of keywords, see previous section on dataset creation) appears at the beginning, in the central part or at the end of each candidate argument.

Similarity with other related / unrelated pairs: the intuition behind this set of features is that related pairs should be more similar to other related pairs than to unrelated ones. For each topic, its merged *related* and *unrelated* pairs are represented as two vectors using a bag-of-words model. Their semantic similarity with the individual pairs in the dataset is computed through cosine similarity and used as a feature.

For classification, we adopt a supervised machine learning approach training Support Vector Machines with radial kernel using LIBSVM [104].

In the *second step* of the classification pipeline, we take in input the outcome of the first step and classify all the pairs of related arguments as support or attack. We rely on a set of surface, sentiment and semantic features inspired by [315] and [314]. We adopt the **Lexical overlap** set of features used also for the first step, to which we add the features described below. In general, we aim at representing more semantic information compared to the previous step, in which lexical features were already quite informative.

Negation: this set of features includes the normalized number of words under the scope of a negation in each argument, and the percentage of overlapping lemmas in the negated phrases of the two arguments.

Keyword embeddings: we use word2vec [320] to extract from each argument a vector representing the keywords of a topic. These vectors are extracted using the continuous bag-of-word algorithm, a windows size of 8 and a vector dimensionality of 50.

Argument entailment: these features indicate if the first argument entails the second one, and vice-versa. To detect the presence of entailment we use the Excitement Open Platform

[295].

Argument sentiment: a set of features based on the sentiment analysis module of the Stanford CoreNLP suite [418] are used to represent the sentiment of each argument, calculated as the average sentiment score of the sentences composing it.

Additional features for lexical overlap, entailment and sentiment are obtained also considering only the subtrees containing a topic keyword instead of the full arguments. The feature vectors are then used to train a SVM with radial kernel with LIBSVM, like in the first classification step.

Evaluation. We test the performance of the classification pipeline using the 1,462 manually annotated pairs with 10-fold cross-validation. The first classification step separates the argument pairs linked by either an *attack* or a *support* relation from the argument pairs with *no relation* (that will be subsequently discarded). The purpose of this first step is to pass the related pairs to the *second step*. Thus, we aim at the highest precision, in order to minimise the number of errors propagated to the second step. Table 4.15 shows the results of the classification for the first step. We choose a configuration that, despite a low recall (0.23), scores a precision of 0.88 on the *attack/support* pairs, providing for the second step a total of 194 argument pairs.

	Unrelated	Attack/Support	Average
Precision	0.56	0.88	0.72
Recall	0.97	0.23	0.60
F1	0.71	0.36	0.65

Table 4.15: Step 1: classification of related / unrelated pairs

The second step classifies the related pairs assigning an *attack* or a *support* label. We provide two evaluations: we report the classifier performance only on the gold *attack* and *support* pairs (Table 4.16), and on the pairs classified as related in the first step (Table 4.17). In this way, we evaluate the classifier also in a real setting, to assess the performance of the end-to-end pipeline.

	Attack	Support	Average
Precision	0.89	0.75	0.82
Recall	0.79	0.86	0.83
F1	0.84	0.80	0.82

Table 4.16: Step 2: classification of *Attack* and *Support* using only gold data.

	Attack	Support	Average
Precision	0.76	0.67	0.72
Recall	0.79	0.86	0.83
F1	0.77	0.75	0.77

Table 4.17: Step 2: classification of *Attack* and *Support* using the output of Step 1.

As expected, accuracy using only gold data is 0.82 (against a random baseline of 0.70), while it drops to 0.72 (against a random baseline of 0.51) in the real-world setting.

We also test a 3-class classifier, with the same set of features used in the two classification steps, obtaining a precision of 0.57. This shows that *support/attack* and *no relation* are better represented by using different sets of features, therefore we opt for two binary classifiers in cascade.

Notice that a comparison of our results with existing approaches to predict argument relations, namely the approach of [425] on persuasive essays, cannot be fairly addressed due to huge differences in the complexity of the used corpus. With their better configuration, [425] obtain an F1 of 0.75 on persuasive essays (that are a very specific kind of texts, human upperbound: macro F1 score of 0.854), and of 0.72 on microtexts [371]. The difference in the task complexity is highlighted also in the inter-annotator agreement. Differently from persuasive essays, where students are requested to put forward arguments in favour and against their viewpoint, in political speeches, candidates often respond to opponents in subtle or implicit ways, avoiding a clear identification of opposing viewpoints.

Error analysis. If we analyse the classifier output at topic level, we observe that overall the performance is consistent across all topics, with the exception of *minimum wage*. In this latter case, the classifier performs much better, with an accuracy of 0.94 in the second step. This is probably due to the fact that Kennedy's and Nixon's statements about minimum wage are very different and the discussion revolves around very concrete items (e.g., the amounts of the minimum wage, the categories that should benefit from it). In other cases, for example disarmament or Cuba, the speakers' wording is very similar and tends to deal with abstract concepts such as freedom, war, peace.

Furthermore, we observe that the classifier yields a better performance with argument pairs by the same person rather than those uttered by different speakers: in the first case, accuracy is 0.86, while in the second one it is 0.79 (Step 2).

Looking at misclassified pairs, we notice very challenging cases, where the presence of linguistic devices like rhetorical questions and repeated negations cannot be correctly captured by our features. Example 11 reports on a pair wrongly classified as *Support* belonging to the *health care* topic:

Example 11

Nixon: *Now, some people might say, Mr. Nixon, won't it be easier just to have the Federal Government take this thing over rather than to have a Federal-State program? Won't it be easier not to bother with private health insurance programs? Yes; it would be a lot simpler, but, my friends, you would destroy the standard of medical care.*

Kennedy: *I don't believe that the American people are going to give their endorsement to the leadership which believes that medical care for our older citizens, financed under social security, is extreme, and I quote Mr. Nixon accurately.*

4.3.3 Visualization and analysis of the argumentation graphs

In this section, we describe how the results of our relation prediction system are then used to construct the argumentation graphs about the debated topics.

Several tools have been proposed to visualize (and then reason upon) argumentation frameworks in the computational argumentation field, e.g., Carneades[19], GRAFIX[20], and ConArg2[21]. However, two main problems arise when trying to use such tools for our purposes: first, they

[19] `http://carneades.github.io/`

[20] `https://www.irit.fr/grafix`

[21] `http://www.dmi.unipg.it/conarg/`

are not tailored to long, natural language snippets (the usual names of arguments in computational argumentation are of the form arg_1), and second, they do not consider the possibility to identify specific argumentation schemes over the provided text. For all these reasons, we decided to rely upon a well-know tool called OVA$^+$ [240], an on-line interface for the manual analysis of natural language arguments. OVA$^+$ grounds its visualization on the Argument Interchange Format (AIF) [108], allowing for the representation of arguments and the possibility to exchange, share and reuse the resulting argument maps. OVA$^+$ handles texts of any type and any length.

The last step of our argument mining pipeline takes in input the labeled pairs returned by the relation prediction module and translates this output to comply with the AIF format. This translation is performed through a script converting the `CSV` input file into `json` file to be load on OVA$^+$ through its online interface.[22] In this mapping, each argument is extracted in order to create an information node (I-node) [108], and then, it is possible to create the associated locution node (L-node) and to specify the name of the speaker. The locution appears, preceded by the name of the participant assigned to it, and edges link the L-node to the I-node via an "Asserting" YA-node, i.e., the illocutionary forces of locutions, as in the Inference Anchoring Theory (IAT) model [75]. Supports or attacks between arguments are represented as follows, always relying upon the standard AIF model. A RA-node (*relation of inference*) should connect two I-nodes. To elicit an attack between two arguments, RA-nodes are changed into CA-nodes, namely *schemes of conflict*. Nodes representing the support and the attack relations are the "Default Inference" and the "Default Conflict" nodes, respectively. Figure 4.6 shows (a portion of) the argumentation graph resulting from the relation prediction step about the topic *minimum wage*, where three I-nodes (i.e., arguments) are involved in one support and one attack relation. The *Asserting* nodes connect each argument with its own source (e.g., K for Kennedy and N for Nixon).

OVA$^+$ allows users to load an analysis, and to visualize it. Given the loaded argumentation graph, the user is supported in analyzing the graph by identifying argumentation schemes [465], and adding further illocutionary forces and relations between the arguments. This final step substantially eases the analysis process by historians and social scientists. Moreover, at the end of the analysis, OVA$^+$ permits to save the final argumentation graph on the user's machine (image or json file).

This graph-based visualization is employed to support political scientists and historians in analysing and modeling political speeches. This proves the usefulness of applying the argumentation mining pipeline over such kind of data: it allows users to automatically identify, among the huge amount of assertions put forward by the candidates in their speeches, the main points on which the candidates disagree (mainly corresponding to the solutions they propose to carry out or their own viewpoints on the previous administrations' effectiveness) or agree (mainly, general-purpose assertions about the country's values to promote).

In the following, we analyze the argumentative structure and content of two of the graphs resulting from the discussed topics (i.e., *minimun wage* and *health care*), highlighting main conflicting arguments among candidates, and other argumentative patterns. Note that this analysis is carried out on the proposed dataset, that contains a subset of all the speeches of the candidates, but gives a clear idea of the kind of analysis that could be performed by scholars on the entirety of the speeches. In general (and this is valid for all the analyzed graphs), we notice that the candidates almost always disagree either on the premises (e.g., who caused the problem to be faced) or on the proposed solutions (the minor claims).

[22]The script and the argumentation graphs about the five topics in our corpus (both gold standard and system's output) are available at `https://dh.fbk.eu/resources/political-argumentation`

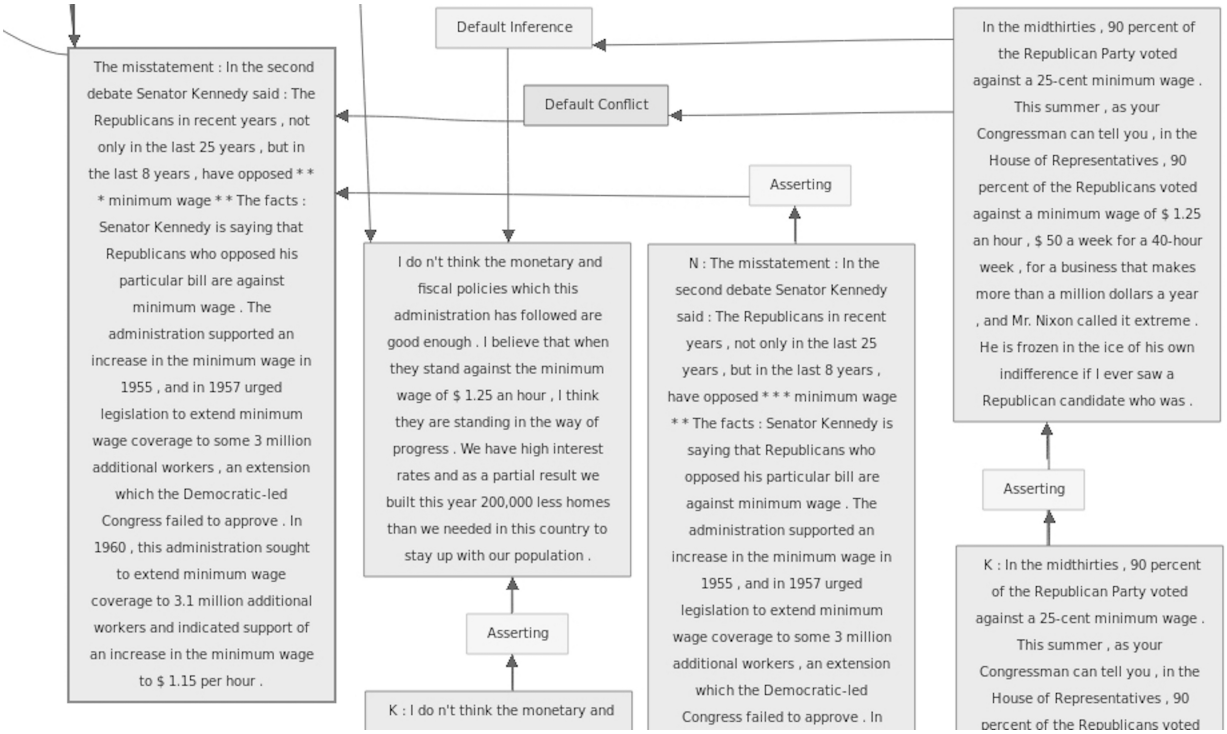


Figure 4.6: The argumentation graph about the topic *minimum wage* visualized through the OVA$^+$ tool.

Minimum wage. A widely discussed topic by both candidates was *minimum wage*, i.e., the bill to set the lowest remuneration that employers may legally pay to workers. It is worth noticing that the argumentation graph for the minimum wage corpus is rather complicated, and it highlights some main controversial issues. The candidates do not agree about the causes of the low minimum wage in 1960 in the US. More precisely, Kennedy attacks the fact that the administration supported an increase in the minimum wage by attacking Nixon's argument *"The misstatement: In the second debate Senator Kennedy said: The Republicans in recent years, not only in the last 25 years, but in the last 8 years, have opposed minimum wage. The facts: [...] The administration supported an increase in the minimum wage in 1955, and in 1957 urged legislation to extend minimum wage coverage to some 3 million additional workers, an extension which the Democratic-led Congress failed to approve. In 1960, this administration sought to extend minimum wage coverage to 3.1 million additional workers and indicated support of an increase in the minimum wage to $1.15 per hour."*. This argument is attacked from different perspectives, leading to a disagreement on the actions the administration carried out in the past years to deal with the minimum wage problem. For instance, as shown in Figure 4.6, Kennedy states that *"In the midthirties, 90 percent of the Republican Party voted against a 25-cent minimum wage. This summer, as your Congressman can tell you, in the House of Representatives, 90 percent of the Republicans voted against a minimum wage of $1.25 an hour, $50 a week for a 40-hour week, for a business that makes more than a million dollars a year, and Mr. Nixon called it extreme. He is frozen in the ice of his own indifference"*. While we may say that this source of disagreement is about the causes of the minimum wage issue, another main source of disagreement is represented by the solutions proposed by the two candidates, which mainly differ regarding the amount of increase of the minimum wage and the coverage of the two respective bills. All these issues become evident with ease in the resulting argumentation graph about the minimum wage topic.

Medical care. The problem of medical care for the elderly was a main problem in 1960, and this topic was widely discussed in the campaign. The resulting argumentation graph highlights some relevant argumentative patterns that are worth analyzing. In general, in the argumentation graphs we are analyzing, the support relation holds between arguments proposed by the same candidate, ensuring in this way a certain degree of coherence in their own argumentation. Interestingly, in the argumentation graph on the topic *medical care*, we can observe that a support relation holds between an argument from Kennedy and one from Nixon, i.e., *"Those forced to rely on surplus food packages should receive a more balanced, nourishing diet. And to meet the pressing problem confronting men past working age, and their families, we must put through an effective program of medical care for the aged under the social security system. The present medical care program will not send one penny to needy persons without further action by the Congress and the State legislatures."* supports *"N: We stand for programs which will provide for increased and better medical care for our citizens, and particularly for those who need it, who are in the older age brackets - and I will discuss that more a little later. We stand for progress in all of these fields, and certainly, as I stand here before you, I am proud to be a part of that platform and of that program"*. These instances of support among candidates mostly concern general issues, i.e., a program of medical care for the elderly is needed.

In summary, our system allows the detection of such argumentation patterns (i.e., topics on which both candidates agree or disagree, topics on which they provide contradictory assertions) and the analysis of how they connect with the other statements asserted in the speeches.

4.4 Mining arguments in 50 years of US presidential campaign debates

Political debates are public interviews where the candidates of elections are requested to confront each other on topics such as unemployment, taxes, and foreign policy. During presidential elections in the US, it is customary for the main candidates of the two largest parties, i.e., the Democratic and the Republican Parties, to engage in a debate around the most controversial issues of the time. Such debates are considered as a *de facto* election process, and in some cases they have nearly decided the outcomes of the elections [120].

Given the importance of these debates and their innate argumentative features, as discussed in the previous section they represent a natural playground for Argument Mining methods. The ability of identifying argumentative components and predicting their relations in such a kind of texts opens the door to cutting-edge tasks like fallacy detection, fact-checking, and counter-argumentation generation.

To be best of our knowledge, none of the few approaches tackling the issue of mining argumentative structures from political debates [278, 312, 155, 459] take on the identification of argument components (i.e., *premises* and *claims*) on a large corpus of political debates. This work fills this gap by *(1)* performing a large-scale annotation study over 50 years of US presidential campaigns from 1960 (Nixon vs. Kennedy) to 2016 (Trump vs. Clinton), resulting in 29k annotated argument components, and *(2)* experimenting with feature-rich SVM learners and neural architectures outperforming standard baselines in Argument Mining. Finally, to ensure full reproducibility of our experiments, we provide all data and source codes under free licenses.

Year	Candidates	T	S	W
1960	Kennedy-Nixon	255	2082	48326
1976	Carter-Ford	270	1874	46444
1980	Anderson-Reagan	200	1141	28765
1984	Mondale-Reagan	365	2376	50126
1988	Bush-Dukakis	484	2599	52780
1992	Bush-Clinton-Perot	929	4057	73688
1996	Clinton-Dole	280	2299	32088
2000	Bush-Gore	564	3225	71852
2000	Cheney-Lieberman	106	835	16395
2004	Bush-Kerry	419	3487	55486
2004	Cheney-Edwards	169	1069	20486
2008	McCain-Obama	505	2829	56379
2012	Obama-Romney	676	2352	62097
2012	Biden-Ryan	425	1252	20785
2016	Clinton-Trump	954	2536	40530
TOT.		6601	34013	676227

Table 4.18: Statistics on the debate transcripts: number of speech turns (T), sentences (S) and words (W).

4.4.1 USElecDeb60To16 dataset creation

The *USElecDeb60To16 v.01* dataset was collected from the website of the Commission on Presidential Debates[23], which provided transcripts of the debates broadcasted on TV and held among the leading candidates for the presidential and vice presidential nominations in the US. USElecDeb60To16 includes the debates starting from Kennedy and Nixon in 1960 until those between Clinton and Trump in 2016. Table 4.18 provides some statistics on the dataset in terms of number of turns in the conversations, of sentences and of words in the transcripts. The unique properties of this dataset are its size (see Table 4.18), its peculiar nature of containing reciprocal discussions (mainly between Democrats and Republicans), and its time line structure. The motivation for creating a new corpus is twofold: *i)* to the best of our knowledge, no other big corpus on political debates annotated at a argument component level for Argument Mining exists, and *ii)* we ensure the reproducibility of the annotation, writing guidelines, inspired from [396, 278], with precise rules for identifying and segmenting argument components (i.e., claims and premises) in political debates.[24]

In the following, we detail the annotation of the argument components through examples from the USElecDeb60To16 dataset.

Claims. Being them the ultimate goal of an argument, in the context of political debates, claims can be a policy advocated by a party or a candidate to be undertaken which needs to be justified in order to be accepted by the audience. In Example 1,[25] Bush is defending the decisions taken by his administration by claiming that his policy has been effective. Claims might also provide judgments about the other candidate or parties (Example 2).

1. **Bush-Kerry, September 30, 2004:**
   BUSH: My administration started what's called the Proliferation Security Initiative. Over 60

[23] `http://www.debates.org`

[24] The *USElecDeb60To16 v.01* dataset and the annotation guidelines are available here: `https://github.com/ElecDeb60To16/Dataset`.

[25] In the examples, claims are marked in **bold**, premises in *Italics* and the component boundaries by [square brackets].

nations involved with disrupting the trans-shipment of information and/or weapons of mass destruction materials. And [**we've been effective**]. [*We busted the A.Q. Khan network. This was a proliferator out of Pakistan that was selling secrets to places like North Korea and Libya*]. [*We convinced Libya to disarm*].

2. **Kennedy-Nixon, September 26, 1960:**
   NIXON: [**I believe the programs that Senator Kennedy advocates will have a tendency to stifle those creative energies**], [**I believe in other words, that his program would lead to the stagnation of the motive power that we need in this country to get progress**].

3. **Kennedy-Nixon, October 13, 1960:**
   NIXON: Senator Kennedy's position and mine completely different on this. [**I favor the present depletion allowance**]. [*I favor it not because I want to make a lot of oil men rich*], but because [*I want to make America rich*]. Why do we have a depletion allowance? Because [*this is the stimulation, the incentive for companies to go out and explore for oil, to develop it*].

Taking a stance towards a controversial subject, or an opinion towards a specific issue is also considered as a claim (e.g., "I've opposed the death penalty during all of my life"). The presence of discourse indicators (e.g., "in my opinion", "I believe") is generally a useful hint in finding claims that state opinions and judgments.

Premises. Premises are assertions made by the debaters for supporting their claims (i.e., reasons or justifications). A type of premise commonly used by candidates is referring to past experience: more experienced candidates exploit this technique to assert that their claims are more relevant than their opponents because of their past experience (Example 4).

4. **Carter-Ford, September 23, 1976:**
   CARTER: [*Well among my other experiences in the past, I've - I've been a nuclear engineer, and did graduate work in this field*]. [**I think I know the - the uh capabilities and limitations of atomic power**].

Statistics are very commonly used as evidence to justify the claims (Example 6). Moreover, premises may be asserted in the form of examples (in such cases, they may contain discourse indicators to introduce examples and justifications, such as "because").

5. **Nixon-Kennedy, September 26, 1960:**
   NIXON: We often hear gross national product discussed and in that respect may I say that [*when we compare the growth in this administration with that of the previous administration that then there was a total growth of eleven percent over seven years*]; [*in this administration there has been a total growth of nineteen percent over seven years*]. [**That shows that there's been more growth in this Administration than in its predecessor**].

6. **Clinton-Dole, October 6, 1996:**
   CLINTON: [*We have ten and a half million more jobs, a faster job growth rate than under any Republican administration since the 1920s*]. [*Wages are going up for the first time in a decade*]. [*We have record numbers of new small businesses*]. [*We have the biggest drop in the number of people in poverty in 27 years*]. [*All groups of people are growing*]. [*We had the biggest drop in income inequality in 27 years in 1995*]. [*The average family's income has gone up over $1600 just since our economic plan passed*]. So [**I think it's clear that we're better off than we were four years ago**].

Three expert annotators defined the annotation guidelines, then three other annotators carried out the annotation task relying on such guidelines. Each transcript has been independently annotated by at least two annotators[26]. 86% of the sentences, which were annotated at least with one component, were tagged with only one argument component, while the remaining 14% with more than one component (7% with both claims and premises).[27] Only 0.6% of the dataset contains cross-sentence annotations (i.e., annotations which are not bound in one sentence). 19 debates have been independently annotated by three annotators to measure the IAA. The observed agreement percentage and IAA at sentence-level (following [424]) are respectively 0.83% and $\kappa = 0.57$ (moderate agreement) for argumentative-non argumentative sentences, and 63% and $\kappa = 0.4$ (fair agreement) for the argument components. Such annotation tasks are very difficult with political debates. In many examples, the choice between a premise and a claim is hard to define. In Example 7, the sentence "the way Senator [...]" is used as a premise for the previous claim, but if observed out of this context, it can be identified as a claim. This justifies the IAA on the argument component annotation.

7. **McCain-Obama, October 15, 2008:**
   OBAMA: [**I disagree with Senator McCain in how to do it**], because [the way Senator McCain has designed his plan, it could be a giveaway to banks if we're buying full price for mortgages that now are worth a lot less]$_?$.

To release a consistent dataset, in the reconciliation phase we computed the IAA of the annotators with two other expert annotators with background in computational linguistics on a sample of 6 debates. In case of disagreement among the first three annotators, the annotation provided by the annotator which showed to be consistently in line with the expert annotators (i.e., with a higher IAA) was included in the released dataset.
After the reconciliation phase, the *USElecDeb60To16* dataset contains the annotation of 29521 argument components (i.e., 16087 claims and 13434 premises). Notice that the number of claims is higher than the number of premises, because in political speeches the candidates make arguments mostly without providing premises for their claims. Moreover, the candidates use longer sentences (more words) to express their premises than their claims.

For our experiments, we split the dataset into train (13894 components), validation (6577 components) and test (9050 components) sets, keeping the same component distribution as in the original dataset.

4.4.2 Experimental setting

We address the argument component detection task as two subsequent classification steps, i.e., the argumentative sentences detection (Task 1), and the argumentative components identification (Task 2). We address both of these classification tasks at the sentence level (e.g., we label a sentence according to the longest component annotated in the sentence).

Methods. For Task 1, we trained both a linear-kernel SVM with stochastic gradient descent learning method using bag of words features only, and a SVM classifier with rbf kernel (python scikit-learn v0.20.1, penalty parameter=10) using the features listed below to distinguish argumentative sentences (i.e., sentences which contain at least one argument component) from the non-argumentative ones. For comparison, we also tested a Neural Network structured with two bidirectional LSTM layers [224] using word embeddings from FastText [245, 321] as the weights for the embedding layer. The output layer determines the class Argumentative/Non-Argumentative for the input sentence. A feed-forward Neural Network was also trained using

[26] We used the Brat annotation tool [428].
[27] A component cannot be both a claim and a premise (see Guidelines).

the same sentence-based features used with the SVM classifier. This network consists of two hidden layers with 64 and 32 neurons for the 1st and 2nd hidden layer, respectively.

As for the component classification step, we applied the same classifiers as for Task 1 (SVM and LSTM). For both tasks, we implemented the majority baseline for argument component classification used in [425].

We considered the following features: tf-idf of each word, NGram (bigrams and trigrams), POS of adverbs, adjectives (used by debaters to stress the correctness of their premises), different tenses of verbs and modal verbs (they often affect the certainty of the assertions, hence would be a hint of facts/non-facts in discerning between argument components), syntactic features (constituency parse trees, dept of the parsing tree), discourse connectives (and their position), NER (debaters often mention party members, former presidents, organizations and dates or numbers like statistics as examples to strengthen the premises for the claims), semantic features (sentiment polarity of the argument component and of its covering sentence [312]).

Evaluation Tables 4.19 and 4.20 present the results obtained on detecting argumentative sentences (Task 1) and classifying argumentative components (Task 2), respectively. Results obtained with linear-kernel SVM significantly outperformed the majority baseline in both tasks. Enriching the feature-set increased the classification performances by 9% on Task 1 using the rbf-kernel SVM, while only by 2.2% on Task 2. Running ablation tests for features analysis, we noticed that the lexical features (tf-idf and NGram features) strongly contribute to performance increase in both tasks. NER features – selected on the assumption that they would have improved the detection of premises as candidate tend to use NERs to provide examples – showed to be more effective in Task 1 only. Sentiment and discourse indicator features did not show to be effective in either classification tasks. Results obtained by LSTM with word-embedding as features in both tasks are comparable to that of the SVM using all the features, showing the efficiency of neural classifiers on AM tasks using less dimensionality for the input data.

Given the complexity of the task, we computed the human upper bound as the average F-score of the annotation agreement between annotators and the gold-standard. It resulted in 0.87 and 0.75 on argumentative vs. non-argumentative, and 0.74 and 0.65 on claims and premises, respectively.

Argumentative sentences are rarely misclassified, which results in high recall on argumentative sentence identification. Some patterns can be identified from the misclassified non-argumentative sentences. One of these patterns appears in very short non-argumentative sentences which contain an argument indicator such as "so", for instance the sentence: "So what should we do?" is classified as argumentative although in the context it is considered a non-argumentative sentence. Since indicators for claims are more numerous in these debates, this misclassification mostly occurs when a claim-indicator is uttered by the candidate in a non-argumentative manner.

In other cases, candidates make final remarks phrasing their speech with a structure similar to argumentative sentences, for example: "I think when you make that decision, it might be well if you would ask yourself, are you better off than you were four years ago?".

Misclassification between claims and premises, instead, is primarily due to the fact that the component classification is highly dependent on the structure of the argument.

4.5 Argument Mining for Healthcare applications

In the healthcare domain, there is an increasing interest in the development of *intelligent* systems able to support and ease clinicians' everyday activities. These systems apply to clinical

Classifier	Class	Precision	Recall	F-Score
Majority baseline	Arg	0.681	1.000	0.810
	None	0.000	0.000	0.000
	Average	0.463	0.681	**0.551**
SVM Linear Kernel BOW	Arg	0.758	0.980	0.855
	None	0.886	0.335	0.486
	Average	0.799	0.774	**0.737**
SVM Rbf Kernel All features	Arg	0.855	0.986	0.916
	None	0.834	0.293	0.433
	Average	0.851	0.853	**0.823**
LSTM network word-embeddings features	Arg	0.882	0.946	0.913
	None	0.668	0.463	0.547
	Average	0.841	0.854	**0.843**
Feed Forward Network All features	Arg	0.885	0.859	0.872
	None	0.471	0.528	0.498
	Average	0.805	0.796	**0.800**

Table 4.19: Classification results on Task 1.

Classifier	Class	Precision	Recall	F-Score
Majority baseline	Claim	0.51	1.00	0.68
	Premise	0.00	0.00	0.00
	Average	0.26	0.51	**0.35**
SVM Linear Kernel BOW	Claim	0.625	0.757	0.685
	Premise	0.682	0.534	0.599
	Average	0.653	0.647	**0.643**
SVM Rbf Kernel All features	Claim	0.631	0.830	0.717
	Premise	0.728	0.484	0.581
	Average	0.678	0.662	**0.651**
LSTM network word-embeddings	Claim	0.848	0.810	0.829
	Premise	0.683	0.739	0.710
	Average	0.673	0.673	**0.673**
Feed Forward Network All features	Claim	0.639	0.697	0.667
	Premise	0.644	0.581	0.611
	Average	0641	0.641	**0.640**

Table 4.20: Classification results on Task 2.

trials, clinical guidelines, and electronic health records, and their solutions range from the automated detection of PICO[28] elements [244] in health records to evidence-based reasoning for decision making [237, 129, 288, 387]. These applications highlight the need of clinicians to be supplied with frameworks able to extract, from the huge quantity of data available for the different diseases and treatments, the exact information they necessitate and to present this information in a structured way, easy to be (possibly semi-automatically) analyzed. Argument(ation) Mining (AM) [371, 279, 92] deals with finding argumentative structures in text. Standard tasks in AM consist in the detection of argument components (i.e., *evidence* and *claims*), and the prediction of the relations (i.e., *attack* and *support*) holding among them. Given its aptness to automatically detect in text those argumentative structures that are at the basis of evidence-based reasoning applications, AM represents a potential valuable contri-

[28]Patient Problem or Population, Intervention, Comparison or Control, and Outcome.

bution in the healthcare domain.

However, despite its natural employment in healthcare applications, only few approaches have applied AM methods to this kind of text [200, 302, 303], and their contribution is limited to the detection of argument components, disregarding the more complex phase of predicting the relations among them. In addition, no huge annotated dataset for AM is available for the healthcare domain. In this work, we cover this gap, and we answer the following research question: *how to define a complete AM pipeline for clinical trials?* To answer this question, we propose a deep bidirectional transformer approach combined with different neural networks to address the AM tasks of component detection and relation prediction in Randomized Controlled Trials, and we evaluate this approach on a new huge corpus of 659 abstracts from the MEDLINE database.

More precisely, the contributions of this work are as follows:

1. We build a new dataset from the MEDLINE database, consisting of 4198 argument components and 2601 argument relations on five different diseases (*neoplasm*, *glaucoma*, *hepatitis*, *diabetes*, *hypertension*)[29];

2. We present a complete AM pipeline for clinical trials relying on deep bidirectional transformers combined with different neural networks, i.e., Long Short-Term Memory (LSTM) networks, Gated Recurrent Unit (GRU) networks, and Conditional Random Fields (CRFs)[30];

3. Our extensive evaluation of various AM architectures (e.g., for persuasive essays) reveals that current approaches are unable to adequately address the challenges raised by medical text and we show that transformer-based approaches outperform these AM pipelines as well as standard baselines.

In the following, we then describe the corpus we built, the methods we employed and the experimental setting. Finally, we report the obtained results and we address an error analysis.

4.5.1 Corpus creation

To address AM on clinical data, we rely on and extend our previous dataset [302], the only available corpus of Randomized Controlled Trial abstracts annotated with the different argument components (evidence, claims and major claims). Such corpus contains the same abstracts used in the corpus of RCT abstracts of [443], that were retrieved directly from PubMed[31] by searching for the disease name and specifying that it has to be a RCT. The first version of the corpus with coarse labels contained 919 argument components (615 evidence and 304 claims) from 159 abstracts comprising 4 different diseases (i.e., *glaucoma*, *hypertension*, *hepatitis b*, *diabetes*).

To obtain more training data, we have extracted from PubMed 500 additional abstracts following Strategy 1 in [443]. We selected *neoplasm*[32] as a topic, assuming that the abstracts would cover experiments over dysfunctions related to different parts of the human body (providing therefore a good generalization as for training instances).

Annotation was started after a training phase, where amongst others the component boundaries were topic of discussion. Gold labels were set after a reconciliation phase, during which

[29]The newly created dataset, called AbstRCT, and the annotation guidelines are available here: `https://gitlab.com/tomaye/abstrct/`

[30]The source code is available here: `https://gitlab.com/tomaye/ecai2020-transformer_based_am`

[31]PubMed (`https://www.ncbi.nlm.nih.gov/pubmed/`) is a free search engine accessing primarily the MEDLINE database on life sciences and biomedical topics.

[32]While neoplasms can either be benign or malignant, the vast majority of articles is about malignant neoplasm (cancer). We stick with *neoplasm* as a term, since this was the MeSH term used for the PubMed query.

Dataset	#Evi	#Claim	#MajCl	#Sup	#Att
Neoplasm	2193	993	93	1763	298
Glaucoma	404	183	7	334	33
Hepatitis	80	27	5	65	1
Diabetes	72	36	11	44	8
Hypertension	59	26	9	53	2
Total	2808	1265	125	2259	342

Table 4.21: Statistics of the extended dataset. Showing the numbers of evidence, claims, major claims, supporting and attacking relations for each disease-based subset, respectively.

the annotators tried to reach an agreement. While the number of annotators vary for the two annotation phases (component and relation annotation), the inter-annotator agreement (IAA) was always calculated with three annotators based on a shared subset of the data. The third annotator was participating in each training and reconciliation phase as well.

In the following, we describe the data annotation process for the argument components in the neoplasm dataset, and for the argumentative relations in the whole dataset. Table 4.21 reports on the statistics of the final dataset.

Annotation of argument components. Following the guidelines for the annotation of argument components in RCT abstracts provided in [302], two annotators with background in computational linguistics[33] carried out the annotation of the 500 abstracts on neoplasm. IAA among the annotators has been calculated on 30 abstracts, resulting in a Fleiss' kappa of 0.72 for argumentative components and 0.68 for the more fine-grained distinction between claims and evidence (meaning substantial agreement for both tasks). Example 1 shows a sample annotated abstract, where claims are written in bold, major claims are highlighted with a dashed underline, and evidence are written in italics.

Claims. In the context of RCT abstracts, a *claim* is a concluding statement made by the author about the outcome of the study. It generally describes the relation of a new treatment (intervention arm) with respect to existing treatments (control arm) and is derived from the described results. *Major claims* are more a general/concluding *claim*, which is supported by more specific claims. The concluding statements do not have to occur at the end of the abstract, and may also occur at the beginning of the text as an introductory *claim*, as in Example 1. Given the negligible occurrences of major claims in our dataset, we merge them with the claims for the classification task.

Evidence. An *evidence* in RCT abstracts is an observation or measurement in the study, which supports or attacks another argument component, usually a *claim*. Those observations comprise side effects and the measured outcome of the intervention and control arm. They are observed facts, and therefore credible without further justifications, as this is the ground truth the argumentation is based on.

Example 1 Extracellular adenosine 5'-triphosphate (ATP) is involved in the regulation of a variety of biologic processes, including neurotransmission, muscle contraction, and liver glucose metabolism, via purinergic receptors. [In nonrandomized studies involving patients with different tumor types including non-small-cell lung cancer (NSCLC), ATP infusion appeared to inhibit

[33] In [201], researchers with different backgrounds (biology, computer science, argumentation pedagogy, and BioNLP) have annotated medical data for an AM task, showing to perform equally well despite their backgrounds.

loss of weight and deterioration of quality of life (QOL) and performance status]. We conducted a randomized clinical trial to evaluate the effects of ATP in patients with advanced NSCLC (stage IIIB or IV). [...] Fifty-eight patients were randomly assigned to receive either 10 intravenous 30-hour ATP infusions, with the infusions given at 2- to 4-week intervals, or no ATP. Outcome parameters were assessed every 4 weeks until 28 weeks. Between-group differences were tested for statistical significance by use of repeated-measures analysis, and reported P values are two-sided. Twenty-eight patients were allocated to receive ATP treatment and 30 received no ATP. *[Mean weight changes per 4-week period were -1.0 kg (95% confidence interval [CI]= 1.5 to -0.5) in the control group and 0.2 kg (95% CI =-0.2 to +0.6) in the ATP group (P=.002)]*$_1$. *[Serum albumin concentration declined by -1.2 g/L (95% CI=-2.0 to -0.4) per 4 weeks in the control group but remained stable (0.0g/L; 95% CI=-0.3 to +0.3) in the ATP group (P =.006)]*$_2$. *[Elbow flexor muscle strength declined by -5.5% (95% CI=-9.6% to -1.4%) per 4 weeks in the control group but remained stable (0.0%; 95% CI=-1.4% to +1.4%) in the ATP group (P=.01)]*$_3$. *[A similar pattern was observed for knee extensor muscles (P =.02)]*$_4$. *[The effects of ATP on body weight, muscle strength, and albumin concentration were especially marked in cachectic patients (P=.0002, P=.0001, and P=. 0001, respectively, for ATP versus no ATP)]*$_5$. [...] This randomized trial demonstrates that **[ATP has beneficial effects on weight, muscle strength, and QOL in patients with advanced NSCLC]$_1$**.

Annotation of argumentative relations. As a next step towards modeling the argumentative structures in the data, it is crucial to annotate the relations, i.e., directed links connecting the components. Those relations are connecting argument components to form the graph like structure of an argument. The relation is a directed link from an outgoing node (i.e., the *source*) to a target node. The nature of the relation can be supporting or attacking, meaning that the source component is justifying or undermining the target component. Links can occur only between certain components: evidence can be connected to either a claim or another evidence, whereas claims can only point to other claims (including major claims). The polarity of the relation (supporting or attacking) does not limit the possibility to what type of component a component can be connected. Theoretically, all types of relations are possible between the allowed combination pairs. Practically, some relations occur rather seldom compared to the frequency of others. The number of outgoing links from a component may exceed one. Furthermore, in rare cases, components cannot be connected at all. This can happen for major claims in the beginning of an abstract, whose function is to point out a related problem, unconnected to the outcome of the study itself.

Attack. A component is attacking another one, if it is *i)* contradicting the proposition of the target component, or *ii)* undercutting its implicit assumption of significance, i.e., stating that the observed effects are not statistically significant. The latter case is shown in Example 2. Here, evidence 1 is attacked by evidence 2, challenging the generality of the prior observation.

Example 2 *[True acupuncture was associated with 0.8 fewer hot flashes per day than sham at 6 weeks,]*$_1$ *[but the difference did not reach statistical significance (95% CI, -0.7 to 2.4; P = .3).]*$_2$

The *partial-attack* is used when the source component is not in full contradiction, but weakening the target component by constraining its proposition. Those can be implicit statements about the significance of the study outcome, which usually occur between two claims (see Example 3). Attacks and partial-attacks are identified with a unique class for the relation classification task.

Example 3 [**SLN biopsy is an effective and well-tolerated procedure.**]$_1$ [**However, its safety should be confirmed by the results of larger randomized trials and meta-analyses.**]$_2$

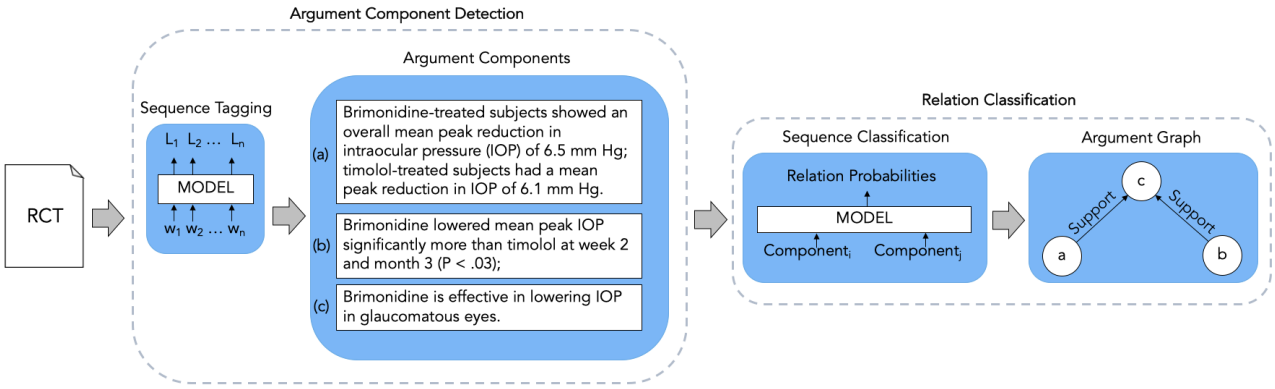


Figure 4.7: Illustration of the full argument mining pipeline on clinical trials.

Support. All statements or observations justifying the proposition of the target component are considered as supporting the target (even if they justify only parts of the target component). In Example 1, all the evidence support claim 1.

We carried out the annotation of argumentative relations over the whole dataset of RCT abstracts, including both the first version of the dataset [302] and the newly collected abstracts on neoplasm. An expert in the medical domain (a pharmacist) validated the annotation guidelines before starting the annotation process. IAA has been calculated on 30 abstracts annotated in parallel by three annotators (the same two annotators that carried out the argument component annotation, plus one additional annotator), resulting in a Fleiss' kappa of 0.62. The annotation of the remaining abstracts was carried out by one of the above mentioned annotators.

4.5.2 The AM pipeline for clinical trials

In this section, we first describe the argument component detection and relation classification tasks, and then we report about the experimental setting to solve these tasks.

Argument Component Detection The first step of the AM pipeline (visualized in Figure 4.7) is the detection of argumentative components and their boundaries. As described above, most of the AM approaches classify the type of component assuming the boundaries of argument components as given. To merge the component classification and boundary detection into one problem, we cast the component detection as sequence tagging task. Following the BIO-tagging scheme, each token should be labeled as either being at the **B**eginning, **I**nside or **O**utside of a component. As we have two component types in AM, this translates into a sequence tagging problem with five labels, i.e., *B-Claim*, *I-Claim*, *B-Evidence*, *I-Evidence* and *Outside*. To model the temporal dynamics of sequence tagging problems, usually Recurrent Neural Networks (RNN) are used. In our experiments, we evaluate different combinations of RNNs with various types of pre-trained word representations. Each embedding method is combined with uni- or bidirectional LSTMs or GRUs with and without a CRF as a last layer. Furthermore, we are the first to do token level classification on AM by fine-tuning different transformer models.

Embeddings. There are two ways to create an input word representation for sequence modelling. One way is to look up the representation from pre-trained embeddings. This static method has the advantage that one does not need to train its own embeddings. However, the vocabulary is limited, and the context of the word is not considered. State-of-the-art embeddings are generated dynamically from the context of the target word based on pre-trained

language models (LM) [4, 144, 375]. In our experiments, we consider both kinds of embeddings. Furthermore, since our data is from the medical domain containing very specific terminology which might not be covered in the vocabulary of general word embeddings, we experiment with different approaches to overcome this problem.

As for the static embeddings, we employ **GloVe** [372] and **extvec** [252] embeddings, which are commonly used and are based on aggregated global word-word co-occurence statistics trained on Wikipedia and the Gigaword 5 corpus. Words are considered to be the smallest unit. Contrary to that, **fastText** [199] and byte-pair embeddings **BPEmb** [219] use subword segments to increase the capability of their vocabulary and might because of that be a better choice for a setting with unusual and specific terminology. Moving to the dynamically generated embeddings, Embeddings from Language Models (**ELMo**) [375] are generating the representation of a word by contextualizing it with the whole input sentence. They use a bidirectional LSTM to independently train a left-to-right and right-to-left character based LM. We use the ELMo model trained on PubMed to have a model which is trained on the type of data we are using. For the same reason, we use the on PubMed trained Contextualized String Embeddings (**FlairPM**) [4], another character-based language model. We compare them directly to embeddings trained on web content, Wikipedia, subtitles and news (**FlairMulti**). The third type of dynamic embedding are Bidirectional Encoder Representations from Transformers (**BERT**) [144]. The language model considers subwords and the position of the word in the sentence to give the final representation of a word.

Transformers can be used as features to an RNN, but also have the possibility to fine-tune the pre-trained model on a target dataset, which we make use of. Beside the original BERT, which is pre-trained on the BooksCorpus and English Wikipedia, there exists multiple other BERT models by now. **BioBERT** [268] is pre-trained on large-scale biomedical corpora outperforming the general BERT model in representative biomedical text mining tasks. The authors initialize the weights with the original BERT model and train on PubMed abstracts and full articles. Therefore, the vocabulary is the same as for the original BERT. Contrary to that, **SciBERT** [44] is trained from scratch with an own vocabulary. While SciBERT is trained on full papers from Semantic Scholar it also contains biomedical data, but to a smaller degree than BioBERT. We chose to use the uncased SciBERT model, meaning that we ignore the capitalization of words. As it was the case for the original BERT, the uncased model of SciBERT performs slightly better for sentence classification tasks than the cased model. Another new model, which outperforms BERT on the General Language Understanding Evaluation (GLUE) benchmark, is **RoBERTa** [284]. There, the BERT pre-training procedure is modified by exchanging static with dynamic masking, using larger byte-pair encoding and batches size, and increasing the size of the dataset.

Relation Classification. After the argument component detection, the next step is to determine which relations hold between the different components (Figure 4.7). We extract valid **BI** tag sequences from the previous step, which are then considered to be the argumentative components of one RCT. Those sequences are phrases and do not necessarily correspond to full sentences. The list of components then serves as input for the relation classification. As explained in Section 4.6, the relation classification task can be tackled with different approaches. We treat it as a sequence classification problem, where the sequence consists of a pair of two components, and the task is to learn the relation between them. For this purpose, we use self-attending transformers, since these models are dominating the benchmarks for tasks which involve classifying the status between two sentences [144]. Treating it as a sequence classification problem gives us two options to model it: *(i)* jointly modelling the relations by classifying all possible argumentative component combinations or *(ii)* predicting possible link

candidates for each entity and then classifying the relation only for plausible entity pairs. In the literature, both methods are represented. Therefore, we decided to evaluate both ways of solving the problem. We experiment with various transformer architectures and compare them with state-of-the-art AM models, i.e., the Tree-LSTM based end-to-end system from Miwa and Bansal [328] as employed by Eger et al. [161], and the multi-objective residual network of Galassi et al. [189]. For option *(i)*, we use bi-directional transformers [144], which consists of an encoder and decoder which themselves consists of multi-head self-attention layer each followed by a fully-connected dense layer. Contrary to the sequence tagging transformer, where each token of the sequence has a representation which is fed into the RNN, for sequence classification a pooled representation of the whole sequence is needed. This representation is passed into a linear layer with a softmax which decodes it into a distribution over the target classes. We treat it as a three class classification problem (*Support*, *Attack* and *NoRelation*). We refer to this type of transformer as **SentClf**. Using this architecture one component can have relations with multiple other components, since each component combination is classified independently. This is not the case in a multiple choice setting (**MultiChoice**), where possible links are predicted taking the other combinations into account and which we employ for *(ii)*. Here, each component (source) is given the list of all the other components as possible target relation candidates and the goal is to determine the most probable candidate as a target component from this list. This problem definition corresponds to the grounded common sense inference problem [494]. To model components which have no outgoing link to other components, we add the *noLink* option to the choice selection. As an encoder for phrase pairs, we evaluate various BERT models which are explained in the transformers section, just as we do for the SentClf task. With respect to the neural transformer architecture, a multiple choice setting means that each choice is represented by a vector $C_i \in \mathbb{R}^H$, where H is the hidden size of the output of an encoder. The trainable weight is a vector $V \in \mathbb{R}^H$ whose dot product with the choice vector C_i is the score of the choice. The probability distribution over all possible choices is given by the softmax, where n is the number of choices:

$$P_i = \frac{e^{V \cdot C_i}}{\sum_{j=1}^{n} e^{V \cdot C_j}} \tag{4.1}$$

The component combination with the highest score of having a link between them is then passed into a linear layer to determine which kind of relation is holding between the two components, i.e., *Attack* or *Support*. The MultiChoice model is trained jointly with two losses, i.e., one for the multiple choice task and one for the relation classification task.

Furthermore, we experimented with linear options for link prediction, such as matrix or tensor factorization. Those methods are widely used on graph data, e.g., knowledge graphs, to discover new links between existing nodes [444]. The matrix or tensor representation of the graph data is decomposed and a model specific scoring function, which assigns a score to each triple[34], is minimized, like a loss function in neural architectures. We experiment by combining those graph-based embeddings and enriching the nodes with linguistic features/embeddings to learn hybrid graph embeddings for relations and discover new links between arguments. The tested linear models are: TuckER [24], TransE [66] and ComplEX [444]. Unfortunately, those models did not learn a meaningful relation representation. We assume this might be due to our relatively small graph data. In the literature, the smallest dataset these models have been experimented on has around 93k triples [142], whereas our dataset has less than 20k.

Experimental Setup. For sequence tagging, each of the above mentioned embeddings were combined with either *(i)* a GRU, *(ii)* a GRU with a CRF, *(iii)* a LSTM, or *(iv)* a LSTM with

[34]A triple consists of a subject (source node), a predicate (labeled edge between nodes) and an object (target node).

		Neoplasm				Glaucoma				Mixed			
Embedding	Model	f_1	F1	C-F1	E-F1	f_1	F1	C-F1	E-F1	f_1	F1	C-F1	E-F1
GloVe	GRU+CRF	.61	.58	.50	.66	.60	.52	.36	.68	.55	.50	.36	.64
extvec	GRU+CRF	.67	.65	.58	.72	.68	.64	.57	.72	.67	.64	.57	.71
fastText(fT)	GRU+CRF	.68	.66	.61	.71	.68	.65	.60	.71	.65	.60	.52	.69
BPEmb	LSTM+CRF	.64	.60	.59	.76	.64	.60	.52	.69	.61	.57	.48	.66
ELMo	LSTM+CRF	.70	.68	.59	.76	.74	.72	.67	.77	.72	.70	.67	.74
BERT	LSTM+CRF	.69	.66	.58	.75	.70	.68	.63	.73	.68	.66	.61	.71
FlairMulti	LSTM+CRF	.66	.63	.53	.72	.58	.55	.50	.60	.52	.50	.44	.56
FlairPM	LSTM+CRF	.70	.68	.60	.75	.74	.72	.69	.75	.70	.68	.64	.72
FlairPM + extvec	GRU+CRF	.68	.65	.54	.74	.74	.72	.67	.77	.68	.66	.60	.72
FlairPM + fT	GRU+CRF	.68	.64	.53	.75	.71	.68	.62	.74	.67	.63	.56	.71
FlairPM + BERT	LSTM+CRF	.70	.69	.61	.76	.71	.70	.67	.73	.68	.67	.62	.72
BERT + fT	LSTM+CRF	.68	.65	.55	.74	.68	.66	.60	.71	.67	.65	.58	.71
ELMo + fT	LSTM+CRF	.71	.68	.59	.77	.74	.72	.69	.77	.72	.70	.65	.75
fine-tuning BERT	dense layer	.82	.60	.69	.83	.77	.55	.63	.80	.80	.57	.65	.83
fine-tuning BERT	GRU+CRF	.89	.85	.78	.90	.89	.86	.76	.89	.90	.88	.81	.91
fine-tuning BioBERT	GRU+CRF	**.90**	.84	.87	.90	**.92**	**.91**	**.93**	.91	**.92**	**.91**	**.91**	.92
fine-tuning SciBERT	GRU+CRF	**.90**	**.87**	**.88**	**.92**	.91	.89	**.93**	.91	.91	.88	.90	**.93**

Table 4.22: Results of the multi-class sequence tagging task are given in micro F1 (f_1) and macro F1 (F1). The binary F1 for claims are reported as C-F1 and for evidence as E-F1. Best scores in each column are marked in bold; significance was tested with a two-sided Wilcoxon signed rank test.

a CRF. Additionally, the best performing static and dynamic embeddings were concatenated and evaluated as if they were one embedding. The *Flair* [4] PyTorch NLP framework version 0.4.1 was used for implementing the sequence tagging task. For BERT, we use the PyTorch implementation of huggingface[35] version 2.3. Hyper parameter tuning was done with hyperopt[36] version 0.1.2. The learning rate was selected from {0.05, 0.1, 0.15, 0.2}, RNN layers {1, 2}, hidden size {32, 64, 128, 256}, dropout {0.1, 0.2, 0.5}, and batch size from {8, 16, 32}. The RNNs were trained over 100 epochs with early stopping and SGD optimizer. For fine-tuning the BERT model, we used the uncased base model with 12 transformer blocks, a hidden size of 768, 12 attention heads, a learning rate of 2e-5 with Adam optimizer for 3 epochs. The same configuration was used for fine-tuning Sci- and BioBERT. For SciBERT, we used the uncased model with the SciBERT vocabulary. For BioBERT, we used version 1.1. For RoBERTa, we increased the number of epochs for fine-tuning to 10, as it was done in the original paper. The best learning rate was 3e-5 on our task. The number of choices for the multiple choice model was 6. Batch size was 8 with a maximum sequence length of 256 subword tokens per input example. We split our neoplasm corpus such that 350 abstracts are assigned to the train, 50 to the development, and 100 to the test set. Additionally, we use the first version of the dataset [302] to create two extra test sets, both comprising 100 abstracts. The first one includes only glaucoma, whereas the second is a mixed set with 20 abstracts of each disease in the dataset (neoplasm, glaucoma, hypertension, hepatitis and diabetes), respectively.

4.5.3 Evaluation

This section presents and discusses the empirical results of our AM pipeline for RCTs.

Sequence Tagging. We show the results for the best performing RNN models and the best performing embedding combinations in Table 4.22. Results are given on all three test sets in micro and macro multi-class F1-score and for claim and evidence, respectively. Comparing the static word embeddings, fastText with a GRU and a CRF is the best performing combination,

[35] `https://github.com/huggingface/transformers`
[36] `https://github.com/hyperopt/hyperopt`

Method	Neoplasm	Glaucoma	Mixed
Tree-LSTM	.37	.44	.39
Residual network	.42	.38	.43
BERT MultiChoice	.58	.56	.55
BioBERT MultiChoice	.61	.58	.57
SciBERT MultiChoice	.63	.59	.60
BERT SentClf	.62	.53	.66
BioBERT SentClf	.64	.58	.61
SciBERT SentClf	**.68**	.62	**.69**
RoBERTa	.67	**.66**	.67

Table 4.23: Results of the relation classification task, given in macro F1-score.

where extvec is only slightly worse and is usually better for evidence classification. For the dynamic embeddings coming from LMs, the ones trained on the medical domain corpus, i.e., FlairPM and ELMo, show similar performances with a macro F1-score of .68 on the neoplasm test set. They have the edge over the non-specialized LMs like BERT with .66 or FlairMulti with .63 macro F1-score. Concatenating static and dynamic embeddings does not bring a notable difference, when taking all test sets into account. Generally, evidence scores are higher than claim scores, leading to the conclusion that claims are more diverse than evidence. The explanation is that, since natural language reports of measurements in clinical trials vary mostly only in the measured parameter and its values, claims can be made about almost everything. Another observation is that the performance of the models trained on neoplasm data do not significantly decrease for test sets on other disease treatments. This fact supports our choice of a more general high level disease type like neoplasm for training the models. The performance for many model combinations even increases on the glaucoma test set. The glaucoma test set comprises only a handful of different glaucoma treatments and is therefore less diversified than the neoplasm or mixed test sets. Looking at the main difference in the results, fine-tuning BERT outperforms all other model combinations, where the version with a GRU and CRF is the best performing model. Fine-tuning without any kind of sequence modelling on top of it results in worse performance. Especially with respect to the validity of BIO sequences, where disproportionately many invalid sequences are generated. This is not useful when extracting the components based on BIO-scheme. Comparing the specialized with the general models, Bio- and SciBERT show a better performance than the general BERT model, where the cased BioBERT tends to be more reliable for the out of domain test data. This is in line with the findings that the cased transformer model works better for tasks like Named Entity Recognition (NER), which is also a sequence tagging task. The difference on our data is marginal: while for NER the casing of a word is relevant, in our task it does not seem to be a sensitive information.

Relation Classification. The results for relation classification are shown in Table 4.23. The numbers are not calculated on gold standard, but show the actual relation classification performance when the components come from the sequence tagging module of the pipeline. We used the best performing sequence tagger, i.e. the fine-tuned SciBERT with a GRU and CRF. We follow previous work on AM [374] and consider the overlap percentage of the components to determine the base if a predicted component matches the annotated component in the gold standard. Since in our data a lot of the components span over 50% or more of a sentence and the exact boundary detection is not always clear, even for human annotators, we consider a predicted and a gold standard component as matched, when at least 75% of the words overlap.

The Tree-LSTM based end-to-end system performed the worst with a F1-score of .37. This can be explained by the positional encoding in the persuasive essay dataset being more relevant than in ours. There, components are likely to link to a neighboring component, whereas in our

dataset the position of a component only partially plays a role, and therefore the distance in the dependency tree is not a meaningful feature. Furthermore, the authors specify that their system does not scale with increasing text length [161]. Especially detailed reports of measurements can make RCT abstracts quite long, such that this system becomes not applicable for this type of data.

The residual network performed better with a F1-score of .42. The main problem here is that it learns a multi-objective for link prediction, relation classification and type classification for source and target component, where the latter classification step is already covered by the sequence tagger and therefore unnecessary at this step.

Similar to sequence tagging, one can see a notable increase in performance when applying a BERT model. Comparing the specialized and general BERT model, the Bio- and SciBERT increase the performance by up to .06 F1-score. Interestingly, RoBERTa delivers comparable results even though it is a model trained on general data. We speculate that parts of the web crawl data which was used to train RoBERTa contain PubMed articles, since they are freely available on the web. Independently of that, RoBERTa shows more reliable results when looking at the performance on the out of domain test sets. While SciBERT as the best performing system on the in-domain test set drops .06 points on the glaucoma test set, RoBERTa stays almost the same and only drops from .67 to .66 F1-score. Looking at the difference between the MultiChoice and SentClf architectures, the SentClf delivers slightly better results, but the drawback is that this technique tends to link components to multiple components. Since most of our components have only one outgoing edge, it creates a lot of false positives, i.e., links which do not exist.

While our dataset consists of only study abstracts for practical reasons, the pipeline can be applied on full text articles as well. Alas, we cannot provide a quantitative analysis on full articles due to missing annotated data. In preliminary experiments on full articles, we have observed a notable increase of false positives in the relation classification, which is the expected consequence of an increased number of components. Furthermore, with the number of components rising in the double-digit range, the multiple-choice architecture loses its predictive power. We leave further investigations to determine the exact limit of this architecture applied on full text articles to future work.

Error Analysis. Common mistakes for the sequence tagger are the invalid BIO sequences. Especially when there are multiple components in one sentence, the tagger tends to mislabel *B-* tokens as *I-* tokens. This is due to the natural imbalance between *B-* and *I-* tokens. Training the sequence tagging without the BIO scheme using only *claim* and *evidence* as labels, poses problems when multiple components are following each other in the text. They would be extracted as one single component instead. This is a common case in concluding sentences at the end of a study, which strikingly often comprise multiple claims. Further experiments could go in the direction of weighted loss functions like focal loss to overcome this problem. Notable mistakes arise for determining the exact component boundaries. Especially in the case of connectives, e.g., *however*, which have sometimes nothing but a conjunctive function, and in other cases signal a constraint of a previous statement. Another mistake is the misclassification of the description of the initial state of the participant groups as an observation of the study and therefore an evidence, e.g., *there were no significant differences in pregnancy-induced hypertension across supplement groups*. In the study abstract these descriptions occur usually relatively close to the actual result description, which means that adding information of the position in the text will not avoid this error. While only some abstracts are structured, the full study report does usually have separated sections. This structure can be exploited when analysing full reports, and in the simplest case one would analyse only the sections of interest.

Concerning link prediction, general components like *the difference was not statistically sig-*

nificant are problematic, since it could be linked to most of the components/outcomes of the trial. Here, a positional distance encoding could be beneficial, since those components are usually connected to the previous component. In general, most of the errors in the MultiChoice architecture were made in the multiple choice part by predicting a wrong link and not at the stage of classifying the relation type. Interestingly, comparing the two domain adapted models, Bio- and SciBERT, there were no regular errors, which allows any conclusion about the advantages or disadvantages of one model. Looking at the confusion matrices, all tested SentClf models show a higher error rate for the *NoRelation* class. Both transformer approaches have in common the problem of dealing with negations and limitations or associating the polarity of a measurement and therefore confusing support and attack.

Example 4 [**more research about the exact components of a VR intervention and choice of outcomes to measure effectiveness is required**]$_{\textbf{source}}$ [**Conducting a pragmatic trial of effectiveness of a VR intervention among cancer survivors is both feasible and acceptable**]$_{\textbf{target}}$

Example 5 [**this did not translate into improved progression-free survival (PFS) or overall survival**]$_{\textbf{source}}$ [**The addition of gemcitabine to carboplatin plus paclitaxel increased treatment burden, reduced PFS time, and did not improve OS in patients with advanced epithelial ovarian cancer**]$_{\textbf{target}}$

Example 4 shows two claims with a limiting/attacking relation, which was wrongly classified as supporting. For Example 5, *not improving progression-free survival (PFS)* corresponds to a *reduced PFS time*, while for other factors reducing the value means it is beneficial and therefore improving some study parameter. Here, the inclusion of external expert knowledge is crucial to learn these fine nuances. The polarity of a measurement cannot be learnt from textual features alone. Especially in the medical domain, there are complex interrelationships which are not often explicitly mentioned and therefore are impossible to capture with a model trained solely on character-based input. Phrases like *increased the blood pressure by X* or *showed no symptom of Y* can connote different messages depending on the context. Future work needs to consider this challenge of incorporating external expert knowledge. While we do not think this is a problem limited to a special domain, we consider it greatly important for understanding and representing medical text.

4.6 Related work

One of the latest advances in artificial argumentation [21] is the so-called *Argument(ation) Mining* [371, 279, 92]. Argument mining consists of two standard tasks: *(i)* the identification of arguments within the text, that may be further split in the detection of argument components (e.g., claims, evidence) and the identification of their textual boundaries. Different methods have been used for this task (e.g., Support Vector Machines (SVMs), Naïve Bayes classifiers, and Neural Networks (NNs)); *(ii)* the prediction of the relations holding between the arguments identified in the first stage. They are used to build the argument graphs, in which the relations connecting the retrieved argumentative components correspond to the edges. Different methods have been employed to address these tasks, from standard SVMs to NNs. AM methods have been applied to heterogeneous types of textual documents, e.g., persuasive essays [425], scientific articles [438], Wikipedia articles [26], political speeches and debates [312], and peer reviews [235]. However, only few approaches [492, 200, 302, 303] focused on automatically detecting argumentative structures from textual documents in the medical domain, such as clinical trials, clinical guidelines, and Electronic Health Records.

Few approaches consider the whole AM pipeline in different application scenarios. In particular, Stab and Gurevych [425] propose a feature-based Integer Linear Programming approach to jointly model argument component types and argumentative relations in persuasive essays. Differently from our data, essays have exactly one major claim each. The authors impose the constraint such that each claim has no more than one parent, while no constraint holds in our case. In contrast with this approach, Eger et al. [161] present neural end-to-end learning methods in AM, which do not require the hand-crafting of features or constraints, using the persuasive essays dataset. They employ TreeLSTM on dependency trees [328] to identify both components and relations between them. They decouple component classification and relation classification, but they are jointly learned, using a dependency parser to calculate the features.

Recent approaches for link prediction rely on pointer networks [384] where a sequence-to-sequence model with attention takes as input argument components and returns the links between them. In these approaches, neither the boundary detection task nor the relation classification one are tackled. Another approach to link prediction relies on structured learning [189]. The authors propose a general approach employing structured multi-objective learning with residual networks, similar to approaches on structured learning on factor graphs [344]. Recently, the argument classification task was addressed with contextualized word embeddings [393]. However, differently from our approach, they assume components are given, and boundary detection is not considered. In line with their work, we experimented with the BERT [144] base model to address parts of the AM pipeline [303] on Randomized Clinical Trials. Contrary to this preliminary work, we now employ and evaluated various contextualized language models and architectures on each task to span the full AM pipeline.

AM on user generated content. The issue of argument detection on Twitter has already been addressed in the literature. [68, 69] address a binary classification task (argument-tweet vs. non argument), as first step of their pipeline. [197] experiments machine learning techniques over a dataset in Greek extracted from social media. They first detect argumentative sentences, and second identify premises and claims. However, none of them is neither interested in distinguishing facts from opinions nor to identify the arguments' sources. An argumentation-based approach is applied to Twitter data to extract opinions in [203], with the aim of detecting conflicting elements in an opinion tree to avoid potentially inconsistent information. Both the goal and the adopted methodology are different from ours.

AM on political speeches and debates. Few approaches apply part of the argument mining pipeline to political debates. Among them, [278] address the problem of argument extraction, and more precisely claim detection, over a corpus based on the 2015 UK political election debates; [160] propose an automatic approach to summarize political debates, starting from a political debates corpus [463]; [156] apply argument mining methods to mine ethos arguments from UK parliamentary debates, while in a follow-up work [155] proposed the *ethos mining* task aiming at detecting ethotic arguments and the relations among the politicians and the parties in the UK Parliament. [338] show how features based on embedding representations can improve discovering various frames in argumentative political speeches. [37] studied the use of semantic frames for modelling argumentation in speakers' discourse. They investigated the impact of argumentation as a influence rank indicator for politicians on the 20 debates for the Republican primary election. Finally, [459] present a dataset composed of the transcripts of televised political debates leading up to the 2016 presidential election in the US, with the addition of the reactions from the social media platform Reddit. The corpus is annotated based on the Inference Anchoring Theory, and not with argument components. Contrary to past works, we create a huge annotated dataset including 39 political debates, and we present a successful attempt to argument component detection on such a big corpus of political debates.

4.7 Conclusions

This chapter focused on my main contributions to the Argument Mining research field. First, two works on AM on user created content (online debates and social media messages) has been presented. In the first work, we have integrated in a combined framework an approach from computational linguistics and a technique for non-monotonic reasoning, with the goal of providing the participants of online debates and forums with a framework supporting their interaction with the application. In particular, the proposed framework helps the participants to have an overview of the debates, understanding which are the accepted arguments at time being. The key contribution of our research is to allow the automatic detection and generation of the abstract arguments from natural language texts. First, we adopt a TE approach to inference because of the kind of (noisy) data present on the Web. TE is used to retrieve and identify the arguments, together with the relation between them: the entailment relation (i.e. inference among two arguments), and the attack relation (i.e. contradiction among two arguments). The arguments and their relations are then sent to the argumentation module which introduces the additional attacks. The argumentation module returns the set of acceptable arguments w.r.t. the chosen semantics. Second, we provide a further step towards a better comprehension of the support and attack notions in bipolar argumentation by evaluating them against *real data* extracted from NL online debates. We point out that the purpose of this work is not to discuss the criticisms advanced against bipolar argumentation [10], nor to support one model over the other. On the contrary, it is intended as a proof of concept of the existing models with respect to real data.

In the second work, we investigated argument mining tasks on Twitter data. The main contribution is twofold: first, we propose one of the very few approaches of argument mining on Twitter, and second, we propose and evaluate two new tasks for argument mining, i.e., facts recognition and source identification. These tasks are particularly relevant when applied to social media data, in line with the open popular challenges of fact-checking and source verification to which these results contribute. Being it a work in progress, several open issues have to be considered as future research. Among them, we are currently extending the dataset of annotated tweets both in terms of annotated tweets per topic, and in terms of addressed topics (e.g., Brexit after the referendum, Trump), in order to have more instances of facts and sources. On such extended dataset, we plan to run experiments using the three modules of the system as a pipeline.

As a second contribution to the AM research field, we have worked on political speeches and debates. In the first work presented in this chapter, an argumentation mining system for relation prediction has been presented and evaluated over a corpus of political speeches from the Nixon-Kennedy U.S. election campaign of 1960. The main advantage of the proposed approach is threefold. First of all, to the best of our knowledge, this is the first approach in argument mining targeting the relation prediction task in monological speeches, where interlocutors do not directly answer to each other. Our approach enables scholars to put together - and more importantly to connect - assertions from the two candidates across the whole political campaign. The output is thus an argumentation graph (one for each topic touched upon in the speeches) summarizing the candidates' own viewpoint and the respective position. Such graphs are intended to support researchers in history, social and political sciences, which must deal with an increasing amount of data in digital form and need ways to automatically extract and analyse argumentation patterns. Second, despite the complexity of the task that constituted a challenge in the annotation phase (and that can be observed in the reported examples), the results we obtained for relation prediction are in line with state-of-the-art systems in argument mining [425]. A third contribution of our work is a resource of 1,462 pairs of natural language arguments annotated with the relations of *support*, *attack*, and *no relation*. In the dataset, each

argument is connected to the source and the date in which the speech was given.

As a follow up work, we investigated the detection of argument components in the US presidential campaign debates: *i)* providing a manually annotated resource of 29k argument components, and *ii)* evaluating feature-rich SVM learners and Neural Networks on such data (achieving ∼90% w.r.t. human performance). We highlighted the strengths (e.g., satisfactory performances on different oratory styles across time and topics) and weaknesses (e.g., no argument boundaries detection on a clause level, the context of the whole debates is not considered).For future work, we plan to *i)* automatically predict relations between argument components in the *USElecDeb60To16* dataset, and *ii)* propose a new task, i.e., *fallacy detection* so that common fallacies in political argumentation [498] can be automatically identified, in line with the work of [207].

As a third contribution to the AM research field, to support clinicians in decision making or in (semi)-automatically filling evidence tables for systematic reviews in evidence-based medicine, we have proposed a complete argument mining pipeline for the healthcare domain. To this aim, we built a novel corpus of healthcare texts (i.e., RCT abstracts) from the MEDLINE database, which are annotated with argumentative components and relations. Indeed, we show that state-of-the-art argument mining systems are unable to satisfactorily tackle the two tasks of argument component detection and relation prediction on this kind of text, given its peculiar features (e.g., component relations spanning across the whole RCT abstract). We expect that our work will have a large impact for clinicians as it is a crucial step towards AI supported clinical deliberation at a large scale.

We employ a sequence tagging approach combining a domain specific BERT model with a GRU and CRF to identify and classify argument components. We cast the relation classification task as a multiple choice problem and compare it with recent transformers for sequence classification. In our extensive evaluation, addressed on a newly AM annotated dataset of RCTs, we investigate the use of different neural transformer architectures and pre-trained models in this pipeline, showing an improvement of the results in comparison with standard baselines and state-of-the-art AM systems. For future work, we will annotate relations across different RCTs to allow reasoning on the resulting argument graphs and clustering of arguments about the same disease. Furthermore, we will investigate different ways to efficiently deal with medical abbreviations and incorporate a distance parameter to overcome the problem that general components talking about limitations are linked to unrelated components far away in the text of the RCT abstract.

Argumentation, emotions and persuasion. As an additional line of works (in the context of the SEEMPAD project, in collaboration with the University of Montreal), we have investigated the correlations between argumentation, emotions and persuasion. Argumentation is often seen as a mechanism to support different forms of reasoning such that decision-making and persuasion, but all these approaches assume a purely rational behavior of the involved actors. However, humans are proved to be have differently, mixing rational and emotional attitudes to guide their actions, and it has been claimed that there exists a strong connection between the argumentation process and the emotions felt by people involved in such process. In our first contributions on this topic [48, 458], we have assessed this claim by means of an experiment: during several debates people's argumentation in plain English is connected and compared to the emotions automatically detected from the participants. Our results show a correspondence between emotions and argumentation elements, e.g., when in the argumentation two opposite opinions are conflicting this is reflected in a negative way on the debaters' emotions.

In follow up works in the same direction, [458, 47] we have explored the role of persuasion, given that in everyday life discussion, people try to persuade each other about the goodness of their viewpoint regarding a certain topic. This persuasion process is usually affected by several

elements, like the ability of the speaker in formulating logical arguments, her confidence with respect to the discussed topic, and the emotional solicitation that certain arguments may cause in the audience. We have compared the effect of using one of the three well-known persuasion strategies (Logos, Ethos and Pathos) in the argumentation process. These strategies are used by a moderator who influences the participants during the debates. We have studied which persuasion strategy is the most effective, and how they vary according to two mental metrics extracted from electroencephalograms: Engagement and workload. Results show that the right hemisphere has the highest engagement when Logos arguments are proposed to participants with Neutral opinion during the debate. We show also that the Logos strategy solicits the highest mental Workload, and the Pathos strategy is the most effective to use in argumentation and to convince the participants.

So far, we have not applied AM methods on the datasets collected through the above mentioned experiments with real users, but that would be an interesting future research direction to explore.

Chapter 5

Cyberbullying and abusive language detection

5.1 Introduction

This chapter summarizes my contributions related to cyberbullying and abusive language detection. While most social media platforms have established user rules that prohibit hate speech, enforcing these rules requires copious manual labor to review every report. Some platforms, such as Facebook, recently increased the number of content moderators. Addressing the challenge of conceiving and implementing automatic tools and approaches to detect inappropriate language or linguistic behavior could accelerate the reviewing process or allocate the human resource to the posts that require close human examination. My research contributions on this topic have been published in several venues:

- Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata (2020). *A multilingual evaluation for online hate speech detection.* ACM journal Transaction in Internet Technology. 20, 2, Article 10 (March 2020), 22 pages [127].

- Pinar Arslan, Michele Corazza, Elena Cabrio, and Serena Villata (2019). *Overwhelmed by negative emotions? maybe you are being cyber-bullied!* In SAC, pages 1061-1063 [17].

- Stefano Menini, Giovanni Moretti, Michele Corazza, Elena Cabrio, Sara Tonelli, and Serena Villata (2019). *A system to monitor cyberbullying based on message classification and social network analysis.* In Proceedings of the Third Workshop on Abusive Language Online, pages 105–110. Association for Computational Linguistics [313].

- Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata (2019). *Cross-platform evaluation for Italian hate speech detection.* In Proceedings of the 6th Italian Conference on Computational Linguistics [125].

- Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata (2019). *Inriafbk drawing attention to offensive language at germeval2019.* In Proceedings of the 15th Conference on Natural Language Processing, KONVENS 2019 [126].

- Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata (2018). *Comparing different supervised approaches to hate speech detection. In Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian.* Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018) [122].

- Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata (2018). *InriaFBK at Germeval 2018: Identifying Offensive Tweets Using Recurrent Neural Networks*. In GermEval 2018 Workshop [123].

The contributions reported in this chapter are the results of several collaborations with the Wimmics team members Serena Villata, Michele Corazza and Pinar Arslan, and - within the CREEP project partners -, mainly with Sara Tonelli and Stefano Menini (FBK, Italy).

As introduced before, the use of social media platforms such as Twitter, Facebook and Instagram has enormously increased the number of online social interactions, connecting billions of users, favouring the exchange of opinions and giving visibility to ideas that would otherwise be ignored by traditional media. However, this has led also to an increase of attacks targeting specific groups of users based on their religion, ethnicity or social status, and individuals often struggle to deal with the consequences of such offenses. This problem affects not only the victims of online abuse, but also stakeholders such as governments and social media platforms. For example, Facebook, Twitter, YouTube and Microsoft have recently signed a code of conduct[1], proposed by the European Union, pledging to review the majority of valid notifications for removal of illegal hate speech in less than 24 hours. Despite the number of approaches recently proposed in the Natural Language Processing research area for detecting these forms of abusive language, the issue of identifying hate speech at scale is still an unsolved problem. To address this issue, we have proposed a robust recurrent neural architecture which is shown to perform in a satisfactory way across different languages, namely English, Italian and German. We have addressed an extensive analysis of the obtained experimental results over the three languages to gain a better understanding of the contribution of the different components employed in the system, both from the architecture and from the feature selection points of view [127].

With such a system, we took part to two shared tasks for hate speech detection: for Italian at the Evalita 2018 evaluation campaign[2][122], and for German at the Germeval 2018[3] and 2019[4] hate speech detection shared task [123, 126], obtaining competitive results w.r.t. state-of-the-art systems (we classified first on one of Evalita's task, and among the top systems in the others).

Moreover, given that most of the available datasets and approaches for hate speech detection proposed so far concern the English language, and even more frequently they target a single social media platform (mainly Twitter), we performed a comparative evaluation on freely available datasets for hate speech detection in Italian, extracted from four different social media platform, i.e. Facebook, Twitter, Instagram and Whatsapp. The reason behind such study is that in low-resource scenarios as the ones we are targeting in the CREEP project it is common to have smaller datasets for specific platforms, raising research questions such as if would it be advisable to combine such platform-dependent datasets to take advantage of training data developed for other platforms. From our study it resulted that the proposed solution of combining platform-dependent datasets in the training phase is beneficial for all platforms but Twitter, for which results obtained by training on tweets only outperform those obtained with a training on the mixed dataset [125].

In the context of the CREEP project, we have implemented a system for the monitoring of cyberbullying phenomena on social media, that aims at supporting supervising persons (e.g., educators) at identifying potential cases of cyberbullying through an intuitive, easy-to-use interface. This displays both the outcome of the hate speech detection system described above and the network in which the messages are exchanged. Supervising persons can therefore monitor the escalation of hateful online exchanges and decide whether to intervene or not.

[1] http://ec.europa.eu/justice/fundamental-rights/files/hate_speech_code_of_conduct_en.pdf
[2] http://www.di.unito.it/~tutreeb/haspeede-evalita18/index.html
[3] https://projects.fzai.h-da.de/iggsa/germeval-2018/
[4] https://projects.fzai.h-da.de/iggsa/

We evaluated the NLP classifier on a set of manually annotated data from Instagram, and detailed the network extraction algorithm starting from high schools in the Manchester area (for English), and in Trentino area (for Italian). However, this is only one possible use case of the system, which can be employed over different kinds of data [313].

This chapter is organized as follows: Section 5.2 describes the robust recurrent neural architecture we propose for the task of hate speech detection. Then, Section 5.3 presents the multi-platform comparative evaluation we carried out for Italian. Section 5.4 describes the system for the monitoring of cyberbullying phenomena on social media. Section 5.5 discusses related work, and conclusions end the chapter.

***NOTE*: This chapter contains examples of language which may be offensive to some readers. They do not represent the views of the authors.**

5.2 A Multilingual evaluation for online hate speech detection

Within the Natural Language Processing (NLP) community, there have been several efforts to deal with the problem of online hate speech detection, since the computational analysis of language can be used to quickly identify offenses and ease the removal of abusive messages. Several workshops [471, 182] and evaluation campaigns [179, 70, 477, 38, 493] have been recently organised to discuss existing approaches to hate speech detection, propose shared tasks and foster the development of benchmarks for system evaluation. These have led to the creation of a number of datasets for hate speech detection in different languages, that have been shared within the NLP research community. Recent advances in deep learning approaches to text classification have then been applied also to deal with this task, achieving for some languages state-of-the-art results [116, 185, 190]. These systems are usually tailored to deal with social media texts by applying pre-processing, using domain-specific embeddings, adding textual features, etc. Given the number of configurations and external resources that have been used by systems for hate speech detection, it is rather difficult to understand what makes a classifier robust for the task, and to identify recommendations on how to pre-process data, what kind of embeddings should be used, etc. To address this issue, after identifying a deep learning architecture that is rather stable and well-performing across different languages, in this section we propose an evaluation of the endowments of several components that are usually employed in the task, namely the type of embeddings, the use of additional features (text-based or emotion-based), the role of hashtag normalisation and that of emojis. We perform our comparative evaluation on English, Italian and German, focusing on freely available Twitter datasets for hate speech detection. Our goal is to identify a set of recommendations to develop hate speech detection systems, possibly going beyond language-specific differences.

5.2.1 Classification framework

Since our goal is to compare the effect of various features, word embeddings and pre-processing techniques on hate speech detection, we use a modular neural architecture for binary classification that is able to support both word-level and message-level features. The components are chosen to support the processing of social-media specific language. The neural architecture and the features are detailed in the following subsections.

Modular Neural Architecture We use a modular neural architecture (see Figure 5.1) in Keras [112]. The architecture that constitutes the base for all the different models uses a single

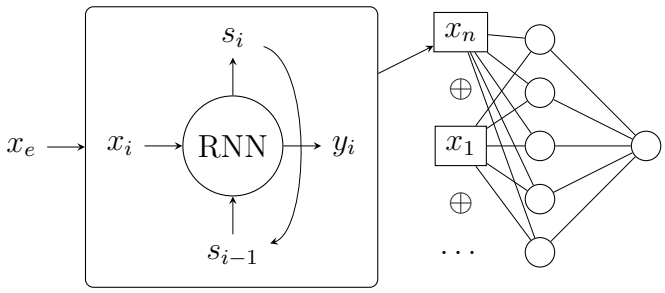


Figure 5.1: The modular neural architecture

feed-forward hidden layer of 100 neurons, with a ReLu activation and a single output with a sigmoid activation. The loss used to train the model is binary cross-entropy. We choose this particular architecture because we used it to participate to two shared tasks for hate speech detection, EVALITA HaSpeeDe 2018 [122] for Italian and Germeval 2018 [124] for German, and it proved to be effective and robust for both languages, also across different social media platforms [125]. In particular, in our original submissions the same architecture was ranked fourth in the Twitter EVALITA subtask (-1.56 F1 compared to the first ranked) and seventh in the Germeval coarse-grained classification task (-2.52 F1 from the top-ranked one).

The architecture is built to support both word-level (i.e. embeddings) and tweet-level features. In particular, we use a recurrent layer to learn an encoding (x_n in Figure 5.1) derived from word embeddings, obtained as the output of the recurrent layer at the last timestep. This encoding gets then concatenated with the other selected features, obtaining a vector of tweet-level features. Since the models derived from using different features are different both in terms of number of parameters and in terms of layers, we decided to keep the size of the hidden layer fixed. This allows us to compare different features, as the latent representation learned by the hidden layer that is ultimately used to classify the tweets has the same size regardless of the number and kind of features.

More formally, given an input represented as the set of features $X = \{x_j | x_j \in X_m\}$, where X_M is the set of all features supported by a model M (see paragraph on features description) and s is the sum of the dimensions of all the features, we compute a function:

$$M(X) = s(W_o H(X) + b_o) \quad W_o \in \mathbb{R}^{1\times 100} \quad H(X) \in \mathbb{R}^{100} \quad b_o \in \mathbb{R}^1$$
$$s(X) = (\sigma(x_1), \ldots, \sigma(x_n)) \quad x \in \mathbb{R}^n$$
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

where W_o and b_o are the learned weights for the output layer and $\sigma(x)$ is the sigmoid activation function (note that in all the models we used $n = 1$ as we only have one binary output), and:

$$H(X) = g(W_h C(X) + b_h) \quad W_h \in \mathbb{R}^{100\times s} \quad C(X) \in \mathbb{R}^s \quad b_h \in \mathbb{R}^{100}$$
$$g(X) = (f(x_1), \ldots, f(x_n)) \quad x \in \mathbb{R}^n$$
$$f(x) = max(0, x) \quad x \in \mathbb{R}$$

where $H(X)$ represents the application of a hidden layer of size 100 and learned weights W_h and b_h and $g(x)$ is the ReLU activation function. Additionally:

$$C(X) = \bigoplus_{x_i \in X} R(x_i)$$

where $\bigoplus$ denotes the concatenation of all vectors along their axes. For example, if we have a set of vectors $X = [x_1, x_2, x_3]$, then:

$$\bigoplus_{x_i \in X} x_i \in \mathbb{R}^{a+b+c} \quad x_1 \in \mathbb{R}^a, x_2 \in \mathbb{R}^b, x_3 \in \mathbb{R}^c$$

Finally:

$$R(x) = \begin{cases} x & \text{if } x \text{ is a tweet-level feature} \\ RNN(x) & \text{if } x \text{ is a word-level feature} \end{cases}$$

where RNN is the function returning the output by a recurrent layer at the last timestep.

Features In our experiments, we use the following features, with the goal of evaluating their impact on a hate speech detection model:

- **Word Embeddings** (x_e in Figure 5.1): multiple word embeddings from various sources have been tested (for a full description of the different embeddings see Section 5.3.1). We evaluate in particular the contribution of word embeddings extracted from social media data, therefore belonging to the specific domain of our classification task, compared with the performance obtained using generic embedding spaces, like Fasttext [62], which are widely used across different NLP tasks because of their good coverage.

- **Emoji embeddings**: emojis are a peculiar element of social media texts. They are often used to emphasize or reverse the literal meaning of a short message, for example in ironic or sarcastic tweets [232]. It is therefore very important for hate speech detection to understand which is the best way to represent them and to include them in the embedding space. We compare different ways to embed emoji information in our classifier: *i)* we use embedding spaces created from social media data, where each emoji is also represented through a word embedding, or *ii)* in case of generic embedding spaces, where emojis are not present, we include emoji embeddings through the alignment of different spaces following the approach presented in [413], or *iii)* in order to cope with the low coverage of emojis, they are replaced by their description in plain text as suggested in [411].

- **Ngrams**: unigrams (x_1 in Figure 5.1) and bigrams derived from the tweets are also included as features. We first tokenize and lemmatize the tweets by using Spacy [228], then normalize the tweet-level ngram occurrence vector by using tf-idf. Our intuition is that these features should capture lexical similarities between training and test data, therefore they should be predictive when training and test set deal with the same type of offenses. Higher-level ngrams are not considered, as we expect them to be very sparse especially in social media, where tweets do not follow standard writing conventions.

- **Social-network specific features**: The character limit imposed by some social media platforms like Twitter affects the style in which messages are written: function words tend to be skipped, texts are very concise while punctuation and uppercase words are used to convey effective messages despite their brevity. Therefore, all these linguistic indicators can be used to identify the presence of hateful messages. We consider in particular the number of hashtags and mentions, the number of exclamation and question marks, the number of emojis, the number of words that are written in uppercase at the tweet-level. These features are then normalized by subtracting their mean and dividing them by their standard deviation.

- **Emotion lexica**: several emotion lexica have been (manually or automatically) created and used in classification tasks to represent the emotional content of a message [334, 333, 41, 426]. While the importance of emotion information to hate speech detection may seem evident [17], it is also true that an embedding space which is large and representative enough of the domain may make additional emotion features redundant. We therefore evaluate the contribution of emotion information using two freely available, multilingual

emotion lexica, namely EmoLex and Hurtlex. Emolex [334, 333] is a large list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive), manually annotated with Amazon's Mechanical Turk. The creators of the lexicon have additionally made available a multilingual version of the resource, that was created by translating each word with Google translate in 2017. We therefore use the German and Italian translations as well as the English one. Using EmoLex, we extract two sentiment-related features and eight emotion-related features for each tweet by summing all the sentiment and emotion scores assigned to the words in a tweet and normalizing them by using tf-idf. The second resource, i.e., Hurtlex [41], is a multilingual lexicon of hate words created starting from the Italian hate lexicon developed by the linguist Tullio De Mauro, organized in 17 categories. It has been expanded through the link to available synset-based computational lexical resources such as MultiWordNet [380] and Babelnet [**?**], and evolved in a multilingual perspective by semi-automatic translation and expert annotation. Since Hurtlex may contain the same word multiple times with different Part-of-Speech tags, we performed a union operation over the categories in order to represent all the categories that a word can belong to, independently of the POS. Using HurtLex, we assign with the same strategy a score for *negative stereotypes*, one for *hate words and slurs* and one for *other insults* to each tweet.

5.2.2 Data and linguistic resources

In the following, we present both the datasets used to train and test our system for English, Italian and German and the word embeddings we have used in our experiments.

English dataset We use the dataset described in [472], containing 16k English tweets manually annotated for hate speech. More precisely, 1,924 are annotated as containing racism, 3,082 as containing sexism, while 10,884 tweets are annotated as not containing offensive language. We merge the sexist and racist tweets in a single class, so that 5,006 tweets are considered as positive instances of hate speech, as in Example 1.

1. Annotation: hateful.
   Since 1/3 of all #Islam believes that people who leave the religion should be murdered where are the moderate Muslim.

Italian dataset We use the Twitter dataset released for the HaSpeeDe (Hate Speech Detection) shared task organized at Evalita 2018, the evaluation campaign for NLP and speech processing tools for Italian[5]. This dataset includes a total amount of 4,000 tweets [70], comprising for each tweet the respective annotation, as can be seen in Example 2. The two classes considered in the annotation are "hateful post" or "not".

2. Annotation: hateful.
   altro che profughi? sono zavorre e tutti uomini (EN: Are they really refugees? they are ballast and all men).

German dataset We use the dataset distributed for the shared task on the Identification of Offensive Language organized at Germeval 2018, a workshop in a series of shared tasks on German processing[6]. The dataset provided for task 1, where offensive comments are to be

[5] `http://www.di.unito.it/~tutreeb/haspeede-evalita18`
[6] `https://www.oeaw.ac.at/ac/konvens2018/workshop/`

Dataset	# hate speech/offensive (%)	# other (%)	# total
English	5,006 (32%)	10,884 (68%)	16,000
Italian	1,296 (32%)	2,704 (68%)	4,000
German	1,688 (34%)	3,321 (66%)	5,009

Table 5.1: Statistics on the datasets

detected from a set of German tweets (binary classification), consists of 5,009 German tweets manually annotated at the message level [477] with the labels "offense" (abusive language, insults, and profane statements) and "other" (i.e. not offensive). More specifically, 1,688 messages are tagged as "offense" (see Example 3), while 3,321 messages as "other".

3. Annotation: Offense.
   @Ralf_Stegner Oman Ralle..dich mag ja immer noch keiner. Du willst das die Hetze gegen dich aufhort? Geh in Rente und verzichte auf die 1/2deiner Pension (EN: @Ralf_Stegner Oman Ralle... still, nobody likes you. You want to stop hate against you? Retire and give up half of your pension).

Table 5.1 summarizes the main statistics on the datasets. The reported values show that, although the datasets have different sizes, the distribution between positive and negative examples is similar. We also manually investigated data samples and the annotation schemes of the English, German and Italian datasets. Although the developers of the English and the Italian corpus focus on hate speech, while the Germeval organisers claim to target offensive language, the kind of messages they annotate as belonging to their respective 'positive' class largely overlap. The targets are different, i.e. the Italian messages focus on immigrants, Muslim and Roma, the English ones on sexist and racial offenses, while the German one has no specific targets, and includes both offensive messags towards groups and towards individuals. However, the types of offenses, both explicit and implicit, including sarcastic messages, rhetorical questions and false claims based on prejudices make them in our view comparable. The only difference is the set of messages labeled as 'Profanity' and included among the 'Offensive' ones in the German dataset, which covers slurs without a specific target. However, they account only for 1.4% messages in this training set.

Word Embeddings In our experiments we test several embeddings, with the goal to compare generic with social media-specific ones. In order to have a high coverage of emojis, we also experiment with aligned embedding spaces obtained by aligning the English, Italian and German ones. Another element we take into account is the access to the binary Fasttext model that originates the embedding space. When using that binary model, it is possible to greatly mitigate the problem of out-of-vocabulary words, since the system is able to provide an embedding for unknown words by using subword unit information [321]. The binary model is often made available together with the standard model when pre-trained embeddings are released. When available, we always use this version. The tested embeddings, summarised in Table 5.2, are the following:

- **Fasttext embeddings for German and Italian**: we use embedding spaces obtained directly from the Fasttext website[7] for German and Italian. In particular, we use the Italian and German embeddings trained on Common Crawl and Wikipedia [199] with size 300. A binary Fasttext model is also available and was therefore used;

[7] `https://fasttext.cc/docs/en/crawl-vectors.html`

- **English Fasttext Crawl embeddings**: English embeddings trained by Fasttext[8] on Common Crawl, with an embedding size of 300. A binary Fasttext model is provided;

- **English Fasttext News embeddings**: English embeddings trained on Wikipedia 2017 using subword information, UMBC web base corpus and statmt.org and released by Fasttext [9], with an embedding size of 300. The available binary Fasttext model was used;

- **Italian Twitter embeddings**: we trained Fasttext embeddings from a sample of Italian tweets [40], with embedding size of 300. We used the binary version of the model;

- **German Twitter embeddings**: trained by Spinning Bytes[10] from a sample of German tweets [113]. We used the model with embeddings of size 300. A binary Fasttext model was not provided, we therefore used the word-based version;

- **English Twitter embeddings**: English Fasttext embeddings from Spinning Bytes[11], trained on an English Twitter sample [113] with an embedding size of 200. Since a binary Fasttext model was not provided, we used the word-based version;

- **Aligned embeddings**: since Fasttext embeddings for Italian and German do not contain emojis, we extend them by aligning them with an English embedding space containing emojis [28], following the alignment approach presented in [413]. All embeddings and the resulting aligned spaces have a size of 300.

EMBEDDINGS	LANGUAGE	ALGORITHM	SIZE	FASTTEXT BINARY MODEL
Fasttext En CCrawl	EN	Fasttext	300	YES
Fasttext En News	EN	Fasttext	300	YES
Twitter English	EN	Fasttext	200	NO
Fasttext It CCrawl & Wiki	IT	Fasttext	300	YES
Twitter Italian	IT	Fasttext	300	YES
Fasttext De CCrawl & Wiki	DE	Fasttext	300	YES
Twitter German	DE	Fasttext	300	NO
Aligned	EN,IT,DE	Fasttext	300	NO

Table 5.2: Overview of the different embeddings used in our experiments

In summary, we were able to use a binary model for all the official Fasttext monolingual datasets and the Italian Twitter embeddings that we trained. For the remaining embedding spaces, we only had access to a dictionary-like structure, that contains the embedding for each word in the vocabulary.

5.2.3 Experiments

In this section, we detail the setup of our experiments, i.e. the pre-processing step, the selection of hyperparameters and the combination of features and configurations tested for each language.

[8] https://fasttext.cc/docs/en/english-vectors.html
[9] https://fasttext.cc/docs/en/english-vectors.html
[10] https://www.spinningbytes.com/resources/wordembeddings/
[11] https://www.spinningbytes.com/resources/wordembeddings/

Preprocessing Since hashtags, user mentions, links to external media and emojis are common in social media interactions, it is necessary to carefully preprocess the data, in order to normalize the text as much as possible while retaining all relevant semantic information. For this reason, we first replace URLs with the word "url" and "@" user mentions with "username" by using regular expressions. Since hashtags often provide important semantic content, we wanted to test how splitting them into single words would impact on the performance of the classifier. To this end, we use the Ekphrasis tool [42] to do hashtag splitting and evaluate the classifier performance with and without splitting. Since the aforementioned tool only supports English, it has been adapted to Italian and German by using language-specific Google ngrams.[12]

Another pre-processing step we evaluate in our experiments is the description of emojis in plain text, that proved to benefit tweet classification [411] but was evaluated so far only on English. In order to map each emoji with a description, we first retrieve an emoji list using the dedicated Python library[13] and replace each emoji with its English description according to the website of the Unicode consortium[14]. We then translate the descriptions using Google Translate and fix any mistakes by hand. In this way we create a list of emojis with the corresponding transcription in three languages (available at `https://github.com/dhfbk/emoji-transcriptions`).

Hyperparameters In order to keep our setting robust across languages, we base our model on a configuration that performed consistently well on all subtasks of Evalita hate speech detection [122], both on Facebook and on Twitter data, even if it was not the best performing configuration on the single tasks. In particular, our model uses no dropout and no batch normalization on the outputs of the hidden layer. Instead, a dropout on the recurrent units of the recurrent layers is used. We select a batch size of 32 for training and a size of 200 for the output (and hidden states) of the recurrent layers. We also test the impact of different recurrent layers, namely long short-memory (LSTM) [224], gated recurrent unit (GRU) [111] and bidirectional LSTM (BiLSTM) [408].

Settings In our experiments, we perform a series of tests on the aforementioned modular neural model, concerning the following aspects:

- For each language, we test the corresponding embeddings.
- We test all possible combination of features: embeddings, unigrams, bigrams, social features, EmoLex and Hurtlex.
- We test three possible recurrent layers, namely LSTM, GRU and Bidirectional LSTM.
- We train models with and without hashtag splitting.
- We test models that replace emojis with their description and models that do not. For Italian and German Fasttext embeddings that do not contain emojis, we also test the model performance after using emoji embeddings resulting from alignment with an English embedding space.

Overall, we compare 1,800 possible configurations for English, 1,080 for Italian and 1,224 for German. The difference is due to the availability of more embedding spaces for English, which increase the amount of possible settings and feature combinations to be tested.

Concerning the dataset splits into training and test instances, for the English dataset - since no standardized split is provided - we randomly selected 60% of the dataset for training, 20%

[12] `http://storage.googleapis.com/books/ngrams/books/datasetsv2.html`
[13] `https://github.com/carpedm20/emoji`
[14] `https://www.unicode.org/emoji/charts/full-emoji-list.html`

for validation and 20% for testing. Since we want our experiments to be reproducible, we use the *train_test_split* function from scikit-learn [370] to shuffle and split the dataset 60%/40%. The remaining 40% was then split in half to obtain the validation and test set, respectively. We use 42 as a seed value for the random number generator used for shuffling.

The German dataset was already randomly split by the GermEval task organizers into training and test set, containing 5,009 and 3,532 messages respectively. For our experiments we keep the same split as proposed in the challenge, but we use 20% of the training set as validation set, obtained by invoking *train_test_split* from scikit learn with 42 as seed. Similarly, the Italian dataset was randomly split by the HaSpeeDe task organizers into training and test set of 3,000 and 1,000 messages respectively. Again, in our experiments we keep the same split as proposed in the challenge, but we used 20% of the training set as validation set applying the same function as for the German dataset.

For each language, the validation test is used to evaluate the classifier performance over 20 training epochs and select the best performing model in terms of macro averaged F1 score. The selected model is then used to evaluate performance on the test set.

5.2.4 Evaluation

In this section, we report a selection of the most relevant results from the pool of settings described in Section 5.2.3. In particular, the first row of Table 5.3, 5.4 and 5.5 reports the best run over all the configurations tested for English, Italian and German respectively, while the other rows show how the best performance changes when modifying one parameter at a time. We also provide an evaluation of the effectiveness of different configurations by comparing the three languages after downsampling the training sets. As comparison we provide a baseline obtained running a SVM (linear kernel) with a bag of word approach using tf-idf as weight.

Multilingual evaluation on the complete datasets

For **English**, the best result (0.823 F1) is obtained using an LSTM network and the Fasttext embeddings trained on Common Crawl. Table 5.3 shows how adding or removing single features from the best configuration affects the result: adding unigram and bigram-based features to the classifier leads to the largest drop in performance, while changing other features the impact is lower. This confirms the findings in [472], in which character n-grams outperform word n-grams in the classification of racial, sexist and not-offensive tweets. Overall we find that, although the best result is obtained using an LSTM network, replacing LSTM with Bi-LSTM keeping the same features achieves similar results, with a difference of F1 of 0.1-0.2% F1. This shows that having both forward and backward information when dealing with tweets is probably not needed because of the limited length of the messages. The use of hashtag normalization to split the hashtags into words improves the system performance in every configuration, increasing the coverage of the embeddings. Overall, the coverage of Fasttext embeddings trained on CommonCrawl is sufficient to deal with Twitter data, therefore adding specific embeddings or pre-processing them is not necessary. Also, the SVM baseline suffers from lower recall compared to the best neural configuration, especially when dealing with the hate category, that has less training instances.

For **Italian**, the best result (0.805 F1) is obtained with a configuration using a LSTM network and the word embeddings we trained on a large corpus of Italian tweets. In Table 5.4 we show to what extent the different features affect the performance obtained with the best configuration. On Italian, differently from English and German, the use of unigrams in addition to word embeddings is beneficial to the classifier performance. The best result is obtained using the emoji transcription, but their impact is not significant (0.805 F1 using them vs. 0.804 not using them). The same trend can be found also with different configurations not reported in the

EMBED.	TEXT FEATS	SOCIAL	EMOT.	EMOJI	NETW.	HASH. SPLIT	F1 NO HATE	F1 HATE	P AVG	R AVG	F1 AVG
Fasttext CCrawl	emb	NO	NO	NO	lstm	YES	0.885	0.760	0.820	0.825	**0.823**
Fasttext CCrawl	emb	NO	NO	NO	lstm	**NO**	0.879	0.745	0.811	0.814	0.812
Fasttext CCrawl	emb	NO	NO	**transcr.**	lstm	YES	0.886	0.756	0.821	0.821	0.821
Fasttext CCrawl	emb	**YES**	NO	NO	lstm	YES	0.883	0.757	0.817	0.823	0.820
Fasttext CCrawl	emb	**YES**	**EMOL**	NO	lstm	YES	0.887	0.751	0.823	0.815	0.819
Fasttext CCrawl	emb	NO	**EMOL**	NO	lstm	YES	0.885	0.751	0.820	0.817	0.818
Fasttext CCrawl	emb	NO	**HURTL**	NO	lstm	YES	0.884	0.744	0.818	0.810	0.814
Fasttext CCrawl	emb	**YES**	**HURTL**	NO	lstm	YES	0.883	0.742	0.816	0.809	0.812
Fasttext CCrawl	**emb+uni+bi**	NO	NO	NO	lstm	YES	0.881	0.719	0.815	0.790	0.800
Fasttext CCrawl	**emb+uni**	NO	NO	NO	lstm	YES	0.871	0.711	0.796	0.787	0.791
Fasttext CCrawl	**emb+bi**	NO	NO	NO	lstm	YES	0.873	0.694	0.800	0.773	0.784
SVM baseline							0.875	0.682	0.808	0.763	0.778

Table 5.3: Best performing configuration on English data (Macro AVG). EMOJI = 'NO' means that no specific processing of emoji was applied

table. Considering all runs with all configurations, the use of embeddings trained on the same domain of the dataset (Italian Tweets) always leads to better results compared with the use of more generic embeddings as the ones from Fasttext (trained on Common Crawl and Wikipedia). Almost all the best performing configurations take advantage of hashtag splitting. BiLSTM performs generally worse than LSTM. Like in the English evaluation, the SVM baseline achieves a remarkably lower performance on the hate class, and shows recall issues.

EMBED.	TEXT FEATS	SOCIAL	EMOT.	EMOJI	NETW.	HASH. SPLIT	F1 NO HATE	F1 HATE	P AVG	R AVG	F1 AVG
Twitter	emb+uni	NO	NO	transcription	lstm	YES	0.867	0.736	0.803	0.806	**0.805**
Twitter	emb+uni	YES	NO	transcription	lstm	YES	0.872	0.737	0.803	0.806	**0.805**
Twitter	emb+uni	NO	NO	**NO**	lstm	YES	0.871	0.736	0.802	0.805	0.804
Twitter	emb+uni	NO	**HURTL**	transcription	lstm	YES	0.867	0.728	0.795	0.800	0.797
Twitter	emb+uni	NO	NO	transcription	lstm	**NO**	0.861	0.727	0.789	0.800	0.794
Twitter	emb+uni	**YES**	**HURTL**	transcription	lstm	YES	0.863	0.723	0.790	0.796	0.793
Twitter	emb+uni	**YES**	**EMOL**	transcription	lstm	YES	0.864	0.718	0.790	0.792	0.791
Twitter	emb+uni	NO	**EMOL**	transcription	lstm	YES	0.858	0.719	0.784	0.794	0.788
Twitter	**emb**	NO	NO	transcription	lstm	YES	0.862	0.697	0.785	0.775	0.779
aligned	emb+uni	NO	NO	**embeddings**	lstm	YES	0.872	0.676	0.809	0.758	0.774
Twitter	**emb+bi**	NO	NO	transcription	lstm	YES	0.860	0.660	0.783	0.747	0.760
Twitter	**emb+uni+bi**	NO	NO	transcription	lstm	YES	0.847	0.690	0.766	0.771	0.768
SVM baseline							0.855	0.593	0.781	0.707	0.724

Table 5.4: Best performing configuration on Italian data (Macro AVG).

Table 5.5 reports the results obtained on **German** data. The best result is achieved with a GRU network, using the standard Fasttext embeddings (trained on Common Crawl and Wikipedia). Similar to English, adopting unigrams and bigrams as feature leads to a decrease in performance (0.05 points F1). Considering all the experiments run on German data, the results confirm that also for this language emoji transcriptions perform better than the emoji vectors obtained through multilingual alignment, but for the best configuration no specific emoji processing is needed. Hashtag splitting, which is included in the best performing configuration for English and Italian, is instead not beneficial to German tweet classification. Our intuition is that, since German is rich in compound words, Ekphrasis hashtag normalization approach based on Google n-grams tends to split terms also when it is not needed. Although social

and emotion features are not used in the best output, they appear to help in most of the other configurations. Also for this language, the SVM baseline achieves a lower recall and less accurate classification on the hate speech class than the neural model.

EMBED.	TEXT FEATS	SOCIAL	EMOT.	EMOJI	NETW.	HASH. SPLIT	F1 NO HATE	F1 HATE	P AVG	R AVG	F1 AVG
Fasttext	emb	NO	NO	NO	GRU	NO	0.829	0.686	0.754	0.762	**0.758**
Fasttext	emb	NO	NO	NO	GRU	**YES**	0.843	0.640	0.765	0.730	0.741
Fasttext	emb	NO	NO	**transcription**	GRU	NO	0.839	0.644	0.758	0.732	0.741
aligned	emb	NO	NO	**embeddings**	GRU	NO	0.835	0.652	0.752	0.738	0.744
Fasttext	emb	**YES**	**HURT**	NO	GRU	NO	0.834	0.671	0.755	0.751	0.753
Fasttext	emb	**YES**	NO	NO	GRU	NO	0.836	0.671	0.759	0.751	0.754
Fasttext	emb	**YES**	**EMOL**	NO	GRU	NO	0.836	0.657	0.755	0.741	0.747
Fasttext	emb	NO	**EMOL**	NO	GRU	NO	0.840	0.655	0.760	0.740	0.748
Fasttext	emb	NO	**HURTL**	NO	GRU	NO	0.843	0.654	0.764	0.739	0.748
Fasttext	**emb+bi**	NO	NO	NO	GRU	NO	0.806	0.606	0.710	0.703	0.706
Fasttext	**emb+uni+bi**	NO	NO	NO	GRU	NO	0.821	0.590	0.726	0.697	0.706
Fasttext	**emb+uni**	NO	NO	NO	GRU	NO	0.819	0.586	0.722	0.694	0.702
SVM baseline							0.807	0.374	0.692	0.597	0.591

Table 5.5: Best performing configuration on German data (Macro AVG).

Beside the aforementioned experiments, we perform an additional evaluation using a character-based RNN. Indeed, character-based representations have been recently used in several NLP tasks including abusive language detection [324] with promising results thanks to their ability to effectively handle rare and unseen words. We use the best performing systems for the three languages, replacing word-based RNN with a character-based one. In order to learn a dense representation for characters, we used a learned embedding layer with size 10. The results of this set of experiments are reported in Table 5.6, and show that using a character-based RNN the performance of the system drops significantly in all three languages compared to word-based RNNs, probably because Fasttext embeddings already account for subword information. We therefore decided not to perform further tests with this configuration.

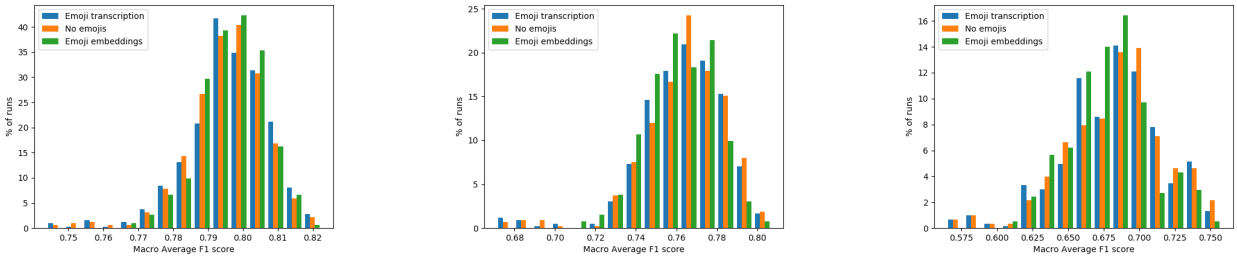
LANG	TEXT FEATS	SOCIAL	EMOTIONS	EMOJI	NETW.	HASH. SPLIT	F1 NO HATE	F1 HATE	P AVG	R AVG	F1 AVG
EN	char	NO	NO	NO	lstm	YES	0.821	0.489	0.697	0.645	0.655
IT	char+uni	NO	NO	transcription	lstm	YES	0.845	0.540	0.763	0.677	0.692
DE	char	NO	NO	NO	GRU	NO	0.771	0.212	0.555	0.524	0.491

Table 5.6: Results of character based RNN using the best configurations for the three languages.

Contribution of social and emotion information

In order to better understand the contribution of specific features or pre-processing steps on all the system runs, we present a comparative evaluation of the classifier performance with or without emoji transcription (in Figure 5.2) and with or without social and emotion features (Figure 5.3). This analysis is done with the goal of focusing not only on the best performing configuration, but also on general trends that could not be included in the previous tables. In particular, we plot the distribution of runs achieving different macro average F1 scores.

Figure 5.2 shows that transcribing emojis yields the best performance for English but not for the two other languages. Nevertheless, this distinction is not clear-cut, since no clear trend can be associated with this feature. More details on the different configurations are shown in Table 5.7, confirming the above findings. Figure 5.3 analyses in a similar way the contribution of social

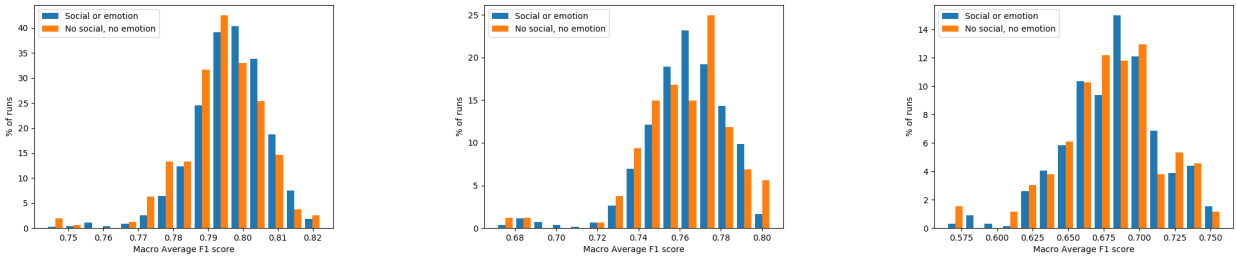


(a) English (total: 1800 runs) (b) Italian (total: 1080 runs) (c) German (total: 1224 runs)

Figure 5.2: Results distribution with and without emoji transcription and using aligned emoji embeddings over the three languages.

Language	Emoji	AVG F1	Max F1	Standard deviation F1	Number of runs
EN	NO	0.796	0.823	0.034	612
EN	YES	0.797	0.821	0.009	576
EN	Transcription	0.796	0.821	0.034	612
IT	NO	0.764	0.804	0.021	468
IT	YES	0.761	0.798	0.016	144
IT	Transcription	0.763	0.805	0.021	468
DE	NO	0.684	0.758	0.034	468
DE	YES	0.678	0.745	0.028	288
DE	Transcription	0.682	0.754	0.034	468

Table 5.7: Mean and Standard deviation of macro averaged F1 scores without any specific processing of emojis ('NO'), using emojis obtained through alignment ('YES') and transcribing them ('TRANSCRIPTION')



(a) English (total: 1800 runs) (b) Italian (total: 1080 runs) (c) German (total: 1224 runs)

Figure 5.3: Results distribution with and without social network and emotion features over the three languages.

Language	Social & emotion features	Mean F1	Max F1	Standard Deviation F1	Number of Runs
EN	NO	0.794	0.823	0.011	300
EN	YES	0.797	0.821	0.010	1500
IT	NO	0.763	0.805	0.020	180
IT	YES	0.763	0.805	0.021	900
DE	NO	0.680	0.758	0.035	204
DE	YES	0.682	0.754	0.033	1020

Table 5.8: Mean and Standard deviation of macro averaged F1 scores with and without social and emotion features

network specific features (i.e. tweet length, punctuation marks, uppercase, etc.) and emotion features (i.e. based on EmoLex and Hurtlex). It shows that, while for English and Italian the best results are obtained without these two groups of features, other runs achieving on average a slightly lower performance make use of this information. For German, the improvement due to social and emotion features appears to be more consistent, even if it does not apply to all runs. Also, the averaged results summarised in Table 5.8 confirm that, like for emojis, the differences are not clear-cut.

Comparing the results across the three languages, we summarize the main findings from the evaluation as follows:

- Using subword information has a positive impact on our task, since it can deal with the high language variability and creativity in the social media domain as well as with typos.
- Creating specific embeddings that cover well the domain of interest is beneficial to the task performance. If possible, a large amount of Twitter data should be collected to create embeddings when dealing with online hate speech classification. If not, pretrained Fasttext embeddings trained on CommonCrawl or similar are recommended, provided that it is possible to access the binary model
- If the above domain-specific embeddings are available, where emojis are also present, our experiments show that it is not needed to pre-process emojis in specific ways (e.g., transcribe, add emoji embeddings through alignment)
- Hashtag normalization is useful to classify hate speech in English and Italian, but current approaches to hashtag splitting may not perform well on languages that are rich in compounds like German, which in turn may affect classification
- Using domain-specific embeddings with a good coverage make emotion lexica redundant in our experiments. The fact that such lexica may be manually or semi-automatically

created does not play a major role in classification performance

- Given the limited length of tweets, LSTM yielded better results than BiLSTM

Multilingual evaluation on downsampled datasets

We perform an additional set of experiments to investigate to what extent the size of the dataset affects the results. Therefore, we downsample both the German and the English datasets to match the size of the Italian Twitter dataset, the smallest one. In order to improve our ability to compare the results, we use the same distribution of labels (hate speech, non hate speech) as the Italian dataset for the two downsampled ones. We then replicate some of the best performing configurations presented in the previous tables, and report the results in Table 5.9. As expected, reducing the training data both for English and for German leads to a drop in performance (from 0.823 F1 to 0.782 for English, from 0.758 F1 to 0.713 for German). On all the runs, the classifier achieves a lower performance on German than on the other two languages, while the results on Italian and English are comparable. Our experiments suggest that German is more challenging to classify, partly because of inherent characteristics of the language (for example the presence of compound words that makes hashtag splitting ineffective), partly because of the way in which the Germeval dataset was built. Namely, the organisers report that they sampled the data starting from specific users and avoiding keyword-based queries, so to obtain the highest possible variability in the offensive language. They also manually checked and enriched the data so to cover all the political spectrum in their offenses, and avoid user overlaps between training and test data. This led to the creation of a very challenging dataset, where lexical overlap between training and test data is limited (therefore unigram and bigram features do not work well) and where hate speech is not associated with specific topics or keywords.

EMBED.	TEXT FEATS	SOCIAL	EMOT.	EMOJI	NETW.	HASH. SPLIT	LANG	F1 NO HATE	F1 HATE	P AVG	R AVG	F1 AVG
Fasttext	emb+uni	NO	NO	transcription	LSTM	YES	EN	0.847	0.683	0.763	0.769	0.765
							IT	0.863	0.739	0.794	0.811	**0.801**
							DE	0.822	0.578	0.726	0.690	0.700
Fasttext	emb+uni	NO	NO	NO	LSTM	YES	EN	0.844	0.696	0.763	0.780	0.770
							IT	0.862	0.723	0.789	0.796	0.793
							DE	0.814	0.563	0.712	0.680	0.689
Fasttext	emb	NO	NO	NO	GRU	NO	EN	0.846	0.701	0.767	0.783	0.773
							IT	0.857	0.708	0.780	0.785	0.783
							DE	0.827	0.598	0.736	0.703	0.713
Fasttext	emb	NO	NO	transcription	LSTM	YES	EN	0.857	0.713	0.780	0.792	**0.785**
							IT	0.849	0.684	0.767	0.766	0.767
							DE	0.824	0.611	0.732	0.710	**0.718**
Fasttext	emb	NO	NO	NO	LSTM	YES	EN	0.853	0.711	0.776	0.791	0.782
							IT	0.837	0.683	0.755	0.767	0.760
							DE	0.830	0.596	0.741	0.702	0.713

Table 5.9: Performance evaluation on data sets of comparable size in English, Italian and German

While our main goal is not to develop a system achieving state-of-the-art results, it is interesting to compare our performance with the best systems dealing with hate speech detection. For Italian and German our approach can be easily compared to other existing classifiers using the same training and test split, since we relied on the official data released in two shared tasks. These results, however, were obtained in the context of the shared task, therefore the authors could not use information about the test set performance as we did. The comparison is still

interesting, but it should be noted that we are reporting the best results on the test set, not on the development set.

On Italian, we observe that our best system configuration achieves state-of-the-art results (F1 0.805). The best performing system in the EVALITA shared task [116] reached 0.800 F1 on the development set using an SVM-based classifier with rich linguistic features, while the best score obtained on the test set (0.799 F1) was yielded by a two-layer BiLSTM in a multi-task learning setting. Similar to our best setting, they also use embeddings extracted from social media data, and observe that using sentiment-based lexica does not increase system performance.

On German, the best performing system participating in Germeval [477] achieved 0.768 F1 [357] and was a stacked ensemble system that combined maximum entropy and random forest classifiers and relied on five groups of features. However, the system performance in 10-fold cross-validation using only the training set reached 0.817 F1. Our best configuration on the task test set yields 0.758 F1 with a much simpler architecture, using only Fasttext and no other features except for word embeddings.

As for English, it is more difficult to draw a similar comparison because the dataset we use [472] was originally annotated with three classes (i.e. racism, sexism and none), thus most systems using the same data perform multiclass classification. Besides, they are run using ten-fold cross-validation like in the original paper [472]. One of the few attempts to distinguish between hate and non-hate speech on the same English data is described in [257], where the authors present a classifier combining word-based CNN and character-based CNN. They report 0.734 F1 on the binary task in ten-fold cross-validation. Other works using the same data set for three-class classification report much higher results (0.783 F1 in [190] using CNN, 0.86 F1 in [257] using a multi-layer perceptron). Interestingly, as shown in [257], multi-class classification seems generally easier than the binary one on this specific data set, since sexist and racist tweets present lexical-based discriminating features that are easy to capture.

Qualitative evaluation

In our experiments we tested more that 1,000 configurations for each language, and it is therefore difficult to manually evaluate and compare the results, since each configuration may make specific mistakes and the distribution of false positives and negatives on the test split would change. In order to gain some insights into the specificity of each language and dataset, however, we focus on the output of the best performing configuration for each language, and we manually check the wrongly classified instances. In most of the cases, it is not possible to assign a category to the mistakes done by the classifier, since the false negative tweets are clearly hateful and the false positive ones are unambiguously non-hateful. These cases are prevalent in all the datasets, so they are independent from the language and also from the dataset size. The opaque mechanisms with which deep learning classifiers assign labels make it difficult to explain why these apparently trivial cases were misclassified, but we plan to exploit information conveyed by attention mechanisms to shed light into this issue [126].

Among the broad mistake categories found across the inspected datasets, there are some cases of implicit abuse. Such messages do not contain abusive words but rather convey their offensive nature through sarcasm, jokes, the usage of negative stereotypes or supposedly objective statements implying some form of offense.

We report few examples of false negatives for the hate speech class below:

4. *It's not about any specific individuals, but about an ideology that will always produce terrorists.*

5. *Molti ancora non vedono, ma quando attraversano un parco, se popolato da immigrati, si tengono stretta la borsa.* (EN: Many do not see it, but when they cross a park populated with immigrants they hold their bag close).

6. *Schau doch Pornos wenn du mehr Redeanteil von Frauen hören willst* (EN: Watch porn if you want to hear more women talk).

We also observe that sentences with a complex syntactic structure, containing for example more than one negation, or questions, are frequent both among the false positives and the false negatives (see Sentence 7, which was wrongly classified as 'Not hate'). The same happens for tweets that contain anaphoric elements that hint at mentions probably present in previous messages, and for tweets which require some form of world knowledge to be understood. In some cases, a link to external media contributed to the hateful meaning of a tweet, as in Sentence 8. However, since we remove urls in the pre-processing step this information was not exploited for classification.

7. *No. You have proven your ignorance here to anyone who isn't as dumb as you. It's there for all to see but you don't know it..*

8. *A quanto pare, il corano si può usare anche per questo. Ma pare non funzioni molto bene.....* *http://t.co/DcOSHfmfxK* (EN: It seems that Quran can be used also for this. But apparently it does not work very well...`http://t.co/DcOSHfmfxK`).

Among false positives, the inspected examples confirm the remarks in [476] concerning the English dataset, and we observe a similar behaviour also for Italian tweets: since these datasets were collected starting from keywords concerning potential hate targets such as women, Roma and Muslims and then extended with not offensive tweets, classifiers tend to associate target mentions to hate speech, even if such messages are not offensive. This phenomenon is less evident on the German data, which indeed was created in a different way, starting from a list of users. Two examples of false positive are reported below. In (9) the message is probably classified as hateful because of the mention of 'Jewish'. In (10) it may depend on the mention of 'migration'.

9. *Fine by me. I had five Jewish friends in college. None ever went to a Synagogue.*

10. *l'immigrazione è un problema x tutti! Ma servono iniziative non comunicati* (EN: Migration is a problem for everybody! But we need initiatives, not press releases).

Finally, we noted few mistakes in the gold standard annotation of the test sets, which were correctly classified by our system.

5.3 Cross-platform evaluation for Italian hate speech detection

Most of the available datasets and approaches for hate speech detection proposed so far concern the English language, and even more frequently they target a single social media platform (mainly Twitter). In low-resource scenarios it is therefore common to have smaller datasets for specific platforms, raising research questions such as: *would it be advisable to combine such platform-dependent datasets to take advantage of training data developed for other platforms? Should such data just be added to the training set or they should be selected in some way? And what happens if training data are available only for one platform and not for the other?*

In this section we address all the above questions focusing on hate speech detection for Italian. Relying on the modular neural architecture that we demonstrated to be rather stable and well-performing across different languages and platforms (in Section 5.2), we perform our comparative evaluation on freely available datasets for hate speech detection in Italian, extracted from four different social media platform, i.e. Facebook, Twitter, Instagram and Whatsapp. In particular, we test the same model while altering only some features and pre-processing aspects. Besides, we use a multi-platform training set but test on data taken from the single platforms. We show that the proposed solution of combining platform-dependent datasets in the training phase is beneficial for all platforms but Twitter, for which results obtained by training on tweets only outperform those obtained with a training on the mixed dataset.

5.3.1 Data and linguistic resources

In the following, we present the datasets used to train and test our system and their annotations. Then, we describe the word embeddings we have used in our experiments.

Twitter dataset released for the HaSpeeDe (Hate Speech Detection) shared task organized at EVALITA 2018 (see dataset description in Section 5.2.2).

Facebook dataset also released for the HaSpeeDe (Hate Speech Detection) shared task. It consists of 4,000 Facebook comments collected from 99 posts crawled from web pages (1,941 negative, and 2,059 positive instances), comprising for each comment the respective annotation, as can be seen in Example 11. The two classes considered in the annotation are "hateful post" or "not".

11. Annotation: hateful.
    Matteo serve un colpo di stato. Qua tra poco dovremo andare in giro tutti armati come in America. (EN: Matteo, we need a coup. Soon we will have to go around armed as in the U.S.).

Whatsapp dataset collected to study pre-teen cyberbullying [420]. Such dataset has been collected through a WhatsApp experimentation with Italian lower secondary school students and contains 10 chats, subsequently annotated according to different dimensions as the roles of the participants (e.g., bully, victim) and the presence of cyberbullying expressions in the message, distinguished between different classes of insults, discrimination, sexual talk and aggressive statements. The annotation is carried out at token level. To create additional training instances for our model, we join subsequent sentences of the same author (to avoid cases in which the user writes one word per message) resulting in 1,640 messages (595 positive instances). We consider as positive instances of hate speech the ones in which at least one token was annotated as a cyberbullying expression, as in Example 12).

12. Annotation: Cyberbulling expression.
    fai schifo, ciccione! (EN: you suck, fat guy).

Instagram dataset includes a total amount of 6,710 messages, which we randomly collected from Instagram focusing on students' profiles (6,510 negative and 200 positive instances) identified through the monitoring system described in [313]. Since no Instagram datasets in Italian were available, and we wanted to include this platform to our study, we manually annotated them as "hateful post" (as in Example 13) or "not".

13. Annotation: hateful.
    Sei una troglodita (EN: you are a caveman).

Platform	Training set	Embeddings	Features	Emoji Transc.	F1 no hate	F1 hate	Macro AVG
Instagram	Multi Plat.	Twitter	Social	Yes	0.984	0.432	0.708
	Single Plat.	Twitter	Social	Yes	0.981	0.424	0.702
Facebook	Multi Plat.	Twitter	Social	Yes	0.773	0.871	0.822
	Single Plat.	Twitter	Social	Yes	0.733	0.892	0.812
WhatsApp	Multi Plat.	Twitter	Social	Yes	0.852	0.739	0.796
	Single Platform	Twitter	Social	Yes	0.814	0.694	0.754
Twitter	Single Plat.	Twitter	Hurtlex	No	0.879	0.717	0.798
	Filtered Multi Plat.	Twitter	Hurtlex	No	0.858	0.720	0.789
	Multi Plat.	Twitter	Hurtlex	No	0.851	0.712	0.782

Table 5.10: Classification results

Word Embeddings In our experiments we test two types of embeddings, with the goal to compare generic with social media-specific ones. In both cases, we rely on Faxttext embeddings [62], since they include both word and subword information, tackling the issue of out-of-vocabulary words, which are very common in social media data (see Section 5.2.2).

5.3.2 System description

Since our goal is to compare the effect of various features, word embeddings, pre-processing techniques on hate speech detection applied to different platforms, we use the modular neural architecture for binary classification presented in Section 5.2.1. For preprocessing methods, we used the ones previously described in Section 5.2.3.

As for features, we evaluate the impact of *i)* word embeddings extracted from social media data, compared with the performance obtained using generic embedding spaces; *ii)* keeping emojis or transcribing them in plain text. To this purpose, we use the official plaintext descriptions of the emojis (from the unicode consortium website), translated to Italian with Google translate and then manually corrected, as a substitute for emojis; *iii)* using a lexicon of hurtful words [41]; *iv)* social media specific features, in particular, the number of hashtags and mentions, the number of exclamation and question marks, the number of emojis, the number of words written in uppercase.

5.3.3 Experimental setup

In order to be able to compare the results obtained while experimenting with different training datasets and features, we used fixed hyperparameters, derived from our best submission at EVALITA 2018 for the cross-platform task that involved training on Facebook data and testing on Twitter. In particular, we used a GRU [111] of size 200 as the recurrent layer and we applied no dropout to the feed-forward layer. Additionally, we used the provided test set for the two Evalita tasks, using 20% of the development set for validation. For Instagram and WhatsApp, since no standard test set is available, we split the whole dataset using 60% of it for training, while the remaining 40% is split in half and used for validation and testing. For this purpose, we use the *train_test_split* function provided by sklearn [370], using 42 as seed for the random number generator. One of our goals was to establish whether merging data from multiple social media platforms can be used to improve performance on single platform test sets. In particular, we used the following datasets for training:

- **Multi-platform**: we merge all the datasets mentioned in Section 5.2.2 for training.

- **Multi-platform filtered by length**: we use the same datasets mentioned before, but only considered instances with a length lower or equal to 280 characters, ignoring URLs and user mentions. This was done to match Twitter length restrictions.

- **Same Platform**: for each of the datasets, we trained and tested the model on data from the same platform.

In addition to the experiments performed on different datasets, we also compare the system performance obtained by using different embeddings. In particular, we train the system by using Italian Fasttext word embeddings trained on CommonCrawl and Wikipedia, and Fasttext word embeddings trained by us on a sample of Italian tweets [40], with an embedding size of 300. We also train our models including either social-media or Hurtlex features. Finally, we compare classification performance with and without emoji transcription.

5.3.4 Results

For each platform, we report in Table 5.10 the best performing configuration considering embedding type, features and emoji transcription. We also report the performance obtained by merging all training data (*Multi-platform*), using only platform-specific training data (*Single platform*) and filtering training instances > 280 characters (*Filtered Multi platform*) when testing on Twitter.

For Instagram, Facebook and Whatsapp, the best performing configuration is identical. They all use emoji transcription, Twitter embeddings and social-specific features. Using multi-platform training data is also helpful, and all the best performing models on the aforementioned datasets use data obtained from multiple sources. However, the only substantial improvement can be observed in the WhatsApp dataset, probably because it is the smallest one, and the classifier benefits from more training data.

The results obtained on the Twitter test set differ from the aforementioned ones in several ways. First of all, the in-domain training set is the best performing one, while the restricted length dataset is slightly better than the non restricted one. These results suggest that learning to detect hate speech on the short length interactions that happen on Twitter does not benefit from using data from other platforms. This effect can be at least partially mitigated by restricting the length of the social interactions considered and retaining only the training instances that are more similar to Twitter ones.

Another remark concerning only Twitter is that Hurtlex is in this case more useful than social network specific features. While the precise cause for this would require more investigation, one possible explanation is the fact that Twitter is known for having a relatively lenient approach to content moderation. This would let more hurtful words slip in, increasing the effectiveness of Hurtlex as a feature, in addition to word embeddings. Additionally, emoji transcription seems to be less useful for Twitter than for other platforms. This might be explained with the fact that the Twitter dataset has relatively less emojis when compared to the others.

One final outtake confirmed by the results is the fact that embeddings trained on social media platforms (in this case Twitter) always outperform general-purpose embeddings. This shows that the language used on social platforms has peculiarities that might not be present in generic corpora, and that it is therefore advisable to use domain-specific resources.

5.4 A system to monitor cyberbullying based on message classification and social network analysis

As introduced before, the presence on social networks like Twitter, Facebook and Instagram is of main importance for teenagers, but this may also lead to undesirable and harmful situations. We refer to these forms of harassment as *cyberbullying*, i.e., 'an aggressive, intentional act carried out by a group or an individual, using electronic forms of contact, repeatedly and over time against a victim who cannot easily defend him or herself' [412]. In online social media, each episode of online activity aimed at offending, menacing, harassing or stalking another person can be classified as a cyberbullying phenomenon. This is connected even with concrete public health issues, since recent studies show that victims are more likely to suffer from psycho-social difficulties and affective disorders [440].

Given its societal impact, the implementation of cyberbullying detection systems, combining abusive language detection and social network analysis, has attracted a lot of attention in the last years [441, 229, 386, 146]. However, the adoption of such systems in real life is not straightforward and their use in a black box scenario is not desirable, given the negative effects misleading analyses could have on potential abusers and victims. A more transparent approach should be adopted, in which cyberbullying identification should be mediated by human judgment.

In this section, we present a system for the monitoring of cyberbullying phenomena on social media. The system aims at supporting supervising persons (e.g., educators) at identifying potential cases of cyberbullying through an intuitive, easy-to-use interface. This displays both the outcome of a hate speech detection system and the network in which the messages are exchanged. Supervising persons can therefore monitor the escalation of hateful online exchanges and decide whether to intervene or not, similar to the workflow introduced in [386]. We evaluate the NLP classifier on a set of manually annotated data from Instagram, and detail the network extraction algorithm starting from 10 Manchester high schools. However, this is only one possible use case of the system, which can be employed over different kinds of data.

5.4.1 Network extraction

Since cyberbullying is by definition a repeated attack towards a specific victim by one or more bullies, we include in the monitoring system an algorithm to identify local communities in social networks and isolate the messages exchanged only within such communities. In this demo, we focus on high-schools, but the approach can be extended to other communities of interest. Our case study concerns the network of Manchester high-school students, and we choose to focus on Instagram, since it is widely used by teenagers of that age.

Reconstructing local communities on Instagram is a challenging task. Indeed, differently from how other social networks operate (e.g., Facebook), Instagram does not provide a page for institutions such as High Schools, that therefore need to be inferred. To overcome this issue, and to identify local communities of students, we proceed in two steps that can be summarised as follow:

- *Expansion stage.* We start from few users that are very likely to be part of the local high school community, and we use them to identify an increasing number of other possible members expanding our network coverage.

- *Pruning stage.* We identify, within the large network, smaller communities of users and we isolate the ones composed by students. For these, we retrieve the exchanged messages in a given period of time (in our case, the ongoing school year), which will be used to identify abusive messages.

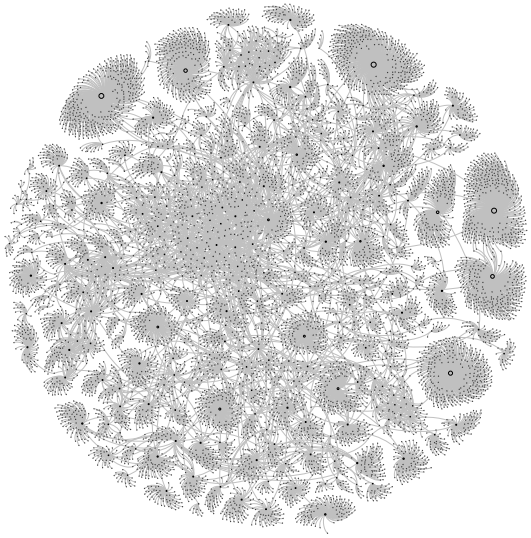


Figure 5.4: Network obtained starting from 10 Manchester schools and expanding +2 layers

5.4.2 Expansion stage

In this stage, we aim to build an inclusive network of people related to local high schools. Since schools do not have an Instagram account, we decide to exploit the geo-tagging of pictures. We manually define a list of 10 high schools from Manchester, and we search for all the photos associated with one of these locations by matching the geo-tagged addresses.

Given that anyone can tag a photo with the address of a school, this stage involves not only actual students, but also their teachers, parents, friends, alumni and so on. The reason to adopt this inclusive approach is that not every student is directly associated with his/her school on Instagram (i.e., by sharing pictures in or of the school), therefore we need to exploit also their contacts with other people directly related to the schools. We restrict our analysis to pictures taken from September 2018 on to focus on the current school year and obtain a network including actual students rather than alumni.

With this approach, we identify a first layer of 756 users, corresponding to the authors of the photos tagged in one of the 10 schools. Starting from these users, we expand our network with a broader second layer of users related to the first ones. We assume that users writing messages to each other are likely to be somehow related, therefore we include in the network all users exchanging comments with the first layer of users in the most recent posts. In this step, we do not consider the connections given by *likes*, since they are prone to introduce noise in the network. With this step we obtain a second layer of 17,810 users that we consider related to the previous ones as they interact with each other in the comments. Using the same strategy, we further expand the network with a third layer of users commenting the contents posted by users in the second layer. It is interesting to notice that in the first layer of users, i.e. the ones directly related to the schools, the groups of users associated with each school are well separated. As soon as we increase the size of the network with additional layers, user groups start to connect to each other through common "friends".

We stop the expansion at a depth of three layers since additional layers would exponentially increase the number of users. At the end of the expansion stage, we gather a list of 544,371 unique users obtained from an exchange of 1,539,292 messages. The resulting network (Figure 5.4) is generated by representing each user as a node, while the exchanged messages correspond to edges. Each edge between two users is weighted according to the number of messages between the two.

5.4.3 Pruning stage

After generating a large network of users starting from the list of schools, the following step consists in pruning the network from *unnecessary nodes* by identifying within the network *smaller communities* of high school students and teenagers. These communities define the scenario in which we want to monitor the possible presence of cyberbullying. To identify local communities, we proceed incrementally dividing the network into smaller portions. For this task, we apply the modularity function in Gephi [59], a hierarchical decomposition algorithm that iteratively optimizes modularity for small communities, aggregating then nodes of the same community to build a new network.

Then, we remove the groups of people falling out of the scope of our investigation by automatically looking for geographical or professional cues in the user biographies. For example, we remove nodes that contain the term *blogger* or *photographer* in the bio, and all the nodes that are only connected to them in the network. This step is done automatically, but we manually check the nodes that have the highest centrality in the network before removing them, so as to ensure that we do not prune nodes of interest for our use case.

We then run again the modularity function to identify communities among the remaining nodes. Finally, we apply another pruning step by looking for other specific cues in the user bios that may identify our young demographic of interest. In this case, we define regular expressions to match the *age*, *year of birth* or *school attended*, reducing the network to a core of 892 nodes (users) and 2,435 edges, with a total of 14,565 messages (Figure 5.5).

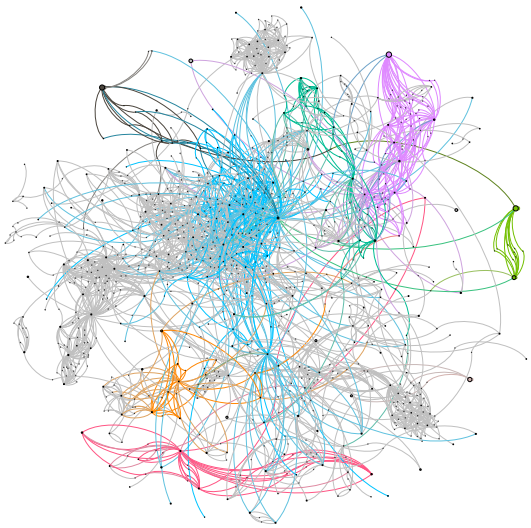


Figure 5.5: Manchester network after pruning

5.4.4 Classification of abusive language

To classify the messages exchanged in the network extracted in the previous step as containing or not abusive language, we use the modular neural architecture for binary classification described in Section 5.2 (same embeddings, same hyperparameters tuning).

5.4.5 Experimental setting and evaluation

Although our use case focuses on Instagram messages, we could not find available datasets from this social network with annotated comments. The widely used dataset used by [230] has indeed annotations at thread level.

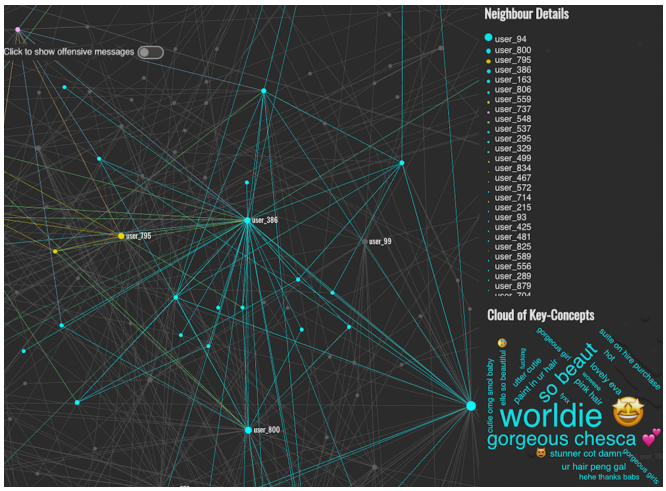


Figure 5.6: Interface view for network exploration

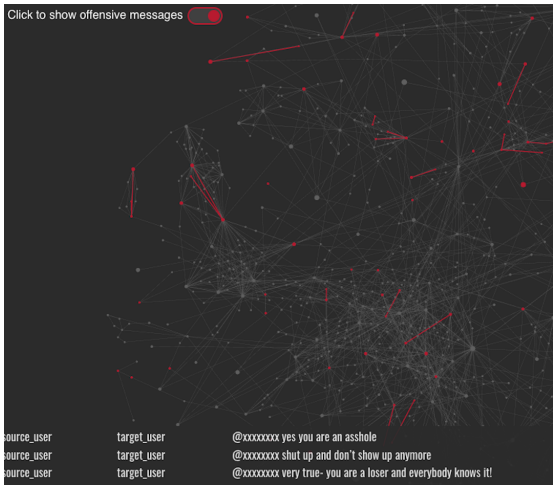


Figure 5.7: Interface view for hate speech monitoring

We therefore train our classification algorithm using the dataset described in [472], containing 16k English tweets manually annotated for hate speech. More precisely, 1,924 are annotated as containing racism, 3,082 as containing sexism, while 10,884 tweets are annotated as not containing offensive language. We merge the sexist and racist tweets in a single class, so that 5,006 tweets are considered as positive instances of hate speech. As a test set, we manually annotate 900 Instagram comments, randomly extracted from the Manchester network, labeling them as hate speech or not. Overall, the test set contains 787 non-offensive and 113 offensive messages.

We preprocess both data sets, given that hashtags, user mentions, links to external media and emojis are common in social media interactions. To normalize the text as much as possible while retaining all relevant semantic information, we first replace URLs with the word "url" and "@" user mentions with "username" by using regular expressions. We also use the Ekphrasis tool [42] to split hashtags into sequences of words, when possible.

The system obtained on the test set a micro-averaged F1 of 0.823. We then run the classifier on all messages extracted for the Manchester network, and make the output available through the platform interface.

5.4.6 Interface

The system[15] relies on a relational database and a tomcat application server. The interface is based on existing javascript libraries such as C3.js (`https://c3js.org`) and Sigma.js (`http://sigmajs.org`).

The platform can be used with two settings: in the first one (Figure 5.6), the Manchester network is displayed, with colors denoting different sub-communities characterised by dense connections. By clicking on a node, the platform displays the cloud of key-concepts automatically extracted from the conversations between the given user and her connections using the KD tool [337]. This view is useful to understand the size and the density of the network and to browse through the topics present in the threads. In the second setting (Figure 5.7), which can be activated by clicking on "Show offensive messages", the communities are all colored in grey, while the system highlights in red the messages classified as offensive by our system. By clicking on red edges it is possible to view the content of the messages classified as offensive, enabling also to check the quality of the classifier. This second view is meant to support educators and stakeholders in monitoring cyberbullying without focusing on single users, but rather keeping

[15] A video of the demo is available at `https://dh.fbk.eu/sites/dh.fbk.eu/files/creepdemo_1.m4v`

an eye on the whole network and zooming in only when hateful exchanges, flagged in red, are escalating.

5.4.7 Discussion

The current system has been designed to support the work of educators in schools, although it is not meant to be open to everyone but only to specific personnel. For example, in Italy there must be one responsible teacher to counter cyberbullying in every school, and access to the system could be given only to that specific person. For the same reason, the system does not show the actual usernames but only placeholders, and the possibility to de-anonymise the network of users could be activated only after cyberbullying phenomena have been identified, and only for the users involved in such cases. Indeed, we want to avoid the use of this kind of platforms for the continuous surveillance of students, and prevent a malicious use of the monitoring platform.

The system relies on public user profiles, and does not have access to content that users want to keep private. This limits the number of cyberbullying cases and hate messages in our use case, where detected abusive language concerns less than 1% of the messages, while a previous study on students' simulated WhatsApp chats around controversial topics reports that 41% of the collected tokens were offensive or abusive [420]. This limitation is particularly relevant when dealing with Instagram, but the workflow presented in this section can be potentially applied to other social networks and chat applications. Another limitation of working with Instagram is the fact that the monitoring cannot happen in real time. In fact, the steps to extract and prune the network require some processing time and cannot be performed on the fly, especially in case of large user networks. We estimate that the time needed to download the data, extract the network, retrieve and classify the messages and upload them in the visualisation tool would be around one week.

5.5 Related Work

5.5.1 Hate speech detection on English data

Given the well-acknowledged rise in the presence of toxic and abusive speech on social media platforms like Twitter and Facebook, an increasing number of approaches has been proposed to detect such a kind of messages in English. Automated systems for the detection of abusive language range from supervised machine learning models built using a combination of manually crafted features such as n-grams [480], syntactic features [345], and linguistic features [488], to more recent neural networks that take word or character sequences from comments and learn abusive patterns without the need for explicit feature engineering. The recent trend of using neural network-based approaches has been particularly evident for English, since several training datasets are available for this language, enabling more data-hungry approaches. Indeed, organisers of the 2019 Semeval task on Offensive Language Identification [493] report that 70% of the participants adopt a deep learning approach. However, also simpler classification systems using logistic regression have been successfully applied to the task [472, 135]. Among the neural network-based approaches, different algorithms have been presented, such as Convolutional Neural Network using pre-trained word2vec embeddings [495], bi-LSTM with attention mechanism [3] and bidirectional Gated Recurrent Unit network [269]. More recently, also the combination of different neural newtorks, capturing both the message content and the Twitter account metadata, has been proposed [185]. In a comparative study of various learning models on the *Hate and Abusive Speech on Twitter* dataset built by Founta et al. [186], Lee et al. [269]

show that, in the classification of tweets as "normal", "spam", "hateful" and "abusive" a bidirectional Gated Recurrent Unit network trained on word-level features is the most accurate model. Instead, in the binary task of offensive language detection, Liu et al. [283] achieve the best performance at Semeval 2019 by fine-tuning a bidirectional encoder representation from transformer [143].

In this work, we propose a robust neural classifier for the hate speech binary classification task which is performing well across different languages (English, Italian and German), and we study the impact of each feature and component on the results across these languages. Our recurrent neural architecture shares some elements with the above approaches, namely the use of a Long Short Term Memory and a Gated Recurrent Unit. Embeddings, textual and social network specific features are employed. As in [257], we do not use metadata related to the social media accounts. The obtained results are compared in a more detailed way in Section 5.2.4.

5.5.2 Hate speech detection on languages different from English

While most approaches to hate speech detection have been proposed for English, other systems have been developed to deal with the task in German, Italian and Spanish, thanks to recent shared tasks. The 2018 GermEval Shared Task on the *Identification of Offensive Language*[16] deals with the detection of offensive comments from a set of German tweets. The tweets have to be classified into the two classes *offense* and *other*, where the *offense* class covers abusive language, insults, as well as profane statements. Different classifiers are used by the participants, ranging from traditional feature-based supervised learning (i.e., SVMs for the top performing system TUWienKBS [357]) to the more recent deep learning methods. Most top performing systems in both shared tasks employed deep learning (e.g., spMMMP [460], uhhLT [474], SaarOffDe [427], InriaFBK [124]). For example, SaarOffDe employs Recurrent Neural Networks and Convolutional Neural Networks produced top scores, while other systems (e.g., spMMMP, uhhLT) employ transfer learning. The usage of ensemble classification seems to often improve the classification approaches (e.g., Potsdam [406], RuG [22], SaarOffDe, TUWienKBS, UdSW [475]). Concerning the features, several systems include a combination of word embeddings, character n-grams and some forms of (task-specific) lexicon. Both the HaUA and the UdSW systems report that high performance scores can be achieved with a classifier solely relying on a lexicon.

In 2018, the first *Hate Speech Detection* (HaSpeeDe) task for Italian has been organized at EVALITA-2018[17]. The task consists in automatically annotating messages from Twitter and Facebook, with a boolean value indicating the presence (or not) of hate speech. Similar to Germeval 2018 submissions, also in this case the participating systems adopt a wide range of approaches, including bi-LSTM [260], SVM [403], ensemble classifiers [382, 23], RNN [184], CNN and GRU [460]. The authors of the best-performing system, ItaliaNLP [116], experiment with three different classification models: one based on linear SVM, another one based on a 1-layer BiLSTM and a newly-introduced one based on a 2-layer BiLSTM which exploits multi-task learning with additional data from the 2016 SENTIPOLC task[18].

Concerning Spanish, the IberEval 2018 edition[19] has proposed the *Aggressiveness Detection* task [99] applied to Mexican Spanish, aiming at providing a classification of aggressive / non-aggressive tweets. A variety of systems is proposed, exploiting content-based (bag of words, word n-grams, term vectors, dictionary words, slang words) and stylistic-based features

[16] https://www.oeaw.ac.at/fileadmin/subsites/academiaecorpora/PDF/GermEval2018_Proceedings.pdf

[17] http://www.evalita.it/2018

[18] http://www.di.unito.it/~tutreeb/sentipolc-evalita16/index.html

[19] https://sites.google.com/view/ibereval-2018

(frequencies, punctuation, POS, Twitter specific elements). Most of the systems rely on neural networks (CNN, LSTM and others). The top ranked team was INGEOTEC [198]: the system is based on MicroT, a text classification approach supported by a lexicon-based model that takes into account the presence of aggressive and affective words, and a model based on the Fasttext representation of texts. More recently, a task for the detection of hate speech against immigrants and women on Twitter has been organised at Semeval 2019 [38], providing an English and Spanish dataset annotated according to the same guidelines. While for both languages a number of neural network approaches has been proposed, the best systems for hateful content detection still rely on SVM and embedding-based features [238, 373, 15].

Looking at the descriptions of the systems participating in the above tasks, as well as at most recent hate speech detection classifiers for English, we observe that deep learning approaches usually share a number of features, such as word embeddings, the use of emotion or sentiment lexica, as well as specific pre-processing steps. Many exploit also other features related to the tweets (e.g., message length, punctuation marks, etc.). Nevertheless, more emphasis is usually put on the architecture, and no insight is given into the role played by variants of the above features and by the selected pre-processing strategy. Also, no attempt to understand differences across different languages has been made.

5.6 Conclusions

Targeting the hate speech detection task in social media messages, in the first part of this chapter we have first described a recurrent neural architecture that is rather stable and well-performing across different languages (i.e., English, German and Italian), and then we have evaluated the contribution of several components that are usually employed in the task, namely the type of embeddings, the use of additional features (text-based or emotion-based), the role of hashtag normalisation and that of emojis. Our comparative evaluation has been carried out on English, Italian and German available Twitter datasets for hate speech detection (annotated as either containing hate speech/offensive language or not). This allowed us to propose a set of findings that could guide researchers in the design of hate speech detection systems, especially for languages different from English.

In the second part of this chapter, we examined the impact of using datasets from multiple platforms in order to classify hate speech on social media. While the results of our experiments successfully demonstrated that using data from multiple sources helps the performance of our model in most cases, the resulting improvement is not always sizeable enough to be useful. Additionally, when dealing with tweets, using data from other social platforms slightly decreases performance, even when we filter the data to contain only short sequences of text. As for future work, further experiments could be performed, by testing all possible combinations of training sources and test sets. This way, we could establish what social platforms share more traits when it comes to hate speech, allowing for better detection systems. At the moment, however, the size of the datasets varies too broadly to allow for a fair comparison, and we would need to extend some of the datasets.

In the last part of the chapter, we presented a platform to monitor cyberbullying phenomena that relies on two components: an algorithm to automatically detect online communities starting from geo-referenced online pictures, and a hate speech classifier. Both components have been combined in a single platform that, through two different views, allows educators to visualise the network of interest and to detect in which sub-communities hate speech is escalating. Although the evaluation has been carried out only on English, the system supports also Italian, and will be showcased in both languages.

Always in this context, as an exploratory and ongoing research activity, we are investigating a zero-shot framework for multilingual hate speech detection. We employed two cross-lingual

language models, i.e., MLM and a new hybrid version of MLM based on emojis (HE–MLM), retraining them to better represent the peculiarity of the language used in social media messages. The second model shows some advantages over the MLM model when used on social media data: first of all, the emojis can be used to convey emotions, that in turn can be correlated with hate speech or offensive content. Secondly, emojis convey similar meaning in the languages that we considered, serving as a common trait between languages during pre-training. Our aim was not to create a system comparable with monolingual state-of-the-art solutions, but to investigate the possibility to use an unsupervised approach for zero-shot cross lingual hate speech detection. As a first step in this direction, we focused on three European languages, for which similar data were available. Despite the high variance observed in the models, we can still observe a clear trend: the HE–MLM models yield better performances than their MLM counterparts, proving that using emojis for pretraining can be useful when dealing with the classification of hate speech on social media text (this work is currently under review).

Chapter 6

Conclusions and Perspectives

To summarize, the research contributions on which I have been working on in the last 9 years have been published in main international conferences and journals of my research communities. They mainly deal with:

- structured information, relations and events extraction from unstructured natural language text, to populate knowledge bases in different application scenarios (e.g., robotics, music information retrieval);
- enhancement of users interactions with the web of data, mapping natural language expressions (e.g., user queries) with concepts and relations in a structured knowledge base (focusing in particular on question answering over linked data);
- the detection of argumentative structures and the prediction of their relations in different textual resources as political debates, medical texts, and social media content (and considering the role of emotions and persuasion);
- abusive language detection, in a multilingual scenario. Taking advantage of both the network analysis and the content of the short-text messages on online platforms, we also addressed the challenge of detecting cyberbullying phenomena, considering the dynamics within a conversation.

In the continuation of my ongoing research work, and in line with the objectives of the Wimmics team, I will keep tackling the four general research questions discussed in this document, in line with the recent trends in Natural Language Processing.

There are a variety of language tasks that, while simple and second-nature to humans, are still very difficult for a machine. New trends in Natural Language Processing systems such as big data and social network analysis, are rapidly emerging and finding application in various domains including - among others - education, travel and tourism (smart cities), and healthcare. Many issues encountered during the development of applications in such domains are addressed as computational tasks, e.g., sentiment analysis, opinion mining, text summarization, dialogue systems, question answering, emotion recognition, revealing an increasing interest in Natural Language Processing and AI, both from the research community and from industry. The NLP field is now seeing a major improvement in the form of a semantically rich representation of words, an accomplishment enabled by the application of neural networks.

In this context, my current and future work is led by the following research question:
How to support the exchange of information and opinions on the Web, helping people and intelligent systems to better understand and evaluate different viewpoints and to promote their own?
Such question breaks down into the following subquestions:

- How to boost the automated identification of natural language arguments on the Web, and the relations among them?
- How to exploit argumentative information to provide advanced search and exploration facilities?
- How to select reliable information from the huge amount of heterogeneous resources available on the Web?
- How to generate argumentation-based explanations and counter-arguments to foster a natural interaction with the users?

Among other use case scenarios, I will keep on focusing on healthcare and politics applications, given that argumentation-based decision making is becoming increasingly prominent in such contexts. In a scenario where doctors debate with each others in order to provide medical diagnoses, and the patients interact with eHealth systems to discuss symptoms and shared experiences, as a first goal we will improve the supervised (Neural Network-based) algorithms we are currently working on, to automatically detect medical arguments from free textual data exchanged in forums and in clinical trials to analyze beside their internal discourse structure (i.e. sentences expressing claims about a topic and related evidences), their relation with respect to other arguments on the same topic (contradiction, support), also their coherence. Similarly, in the automated analysis of political speeches and debates, I plan to automatically predict relations between argument components (intra- and inter-arguments), and propose a new task, i.e., *fallacy detection* so that common fallacies in political argumentation [498] can be automatically identified. In this context, our goal is to support the users in making decisions about how to proceed with the discussion, support the patients and citizens in understanding the possible acceptability of the arguments and their evidences, with regard to their consistency and the sources proposing them, and improve the comprehension of the ongoing discussions.

As a second goal, I will tackle the challenge of exploiting argumentative information to provide advanced search and exploration facilities. The abundance of information found in online communities debates is largely left unexploited by current applications, Web services or search engines. My goal is therefore to develop applications with advanced forms of search and exploration capabilities, which will allow novel and higher-level methods of navigation, based on the above described annotations associated with arguments rather than just hyperlinks.

For this task we will extend current state-of-the art question answering system to address much more complex questions, so that arguments will be returned to support the response of intelligent systems to user information needs expressed through natural language questions. Ranking methods will be applied to return the answers according to user preferences and other criteria (e.g., topical and contextual relevance), and the results of reasoning (new arguments generation using information from existing ones). To improve the answer quality, the most representative among the relevant arguments will be identified, i.e., those that exhibit some diversity while covering the user's information need (considering various aspects, such as content, topic, context, intended audience and relations). Finally, we will study navigation means for exploring the structure and interconnections of the returned arguments, leading to other potentially interesting arguments and providing an overview of the argumentative graph. Summarization techniques will be applied to summarize long debates into smaller ones, whereas elaboration will make an argument more understandable for the specific user and/or query context, by adding information extracted from other arguments, or from resources. This analysis will enable the user to get a global, high-level view of the various opinions on a topic, where justified arguments are supported by facts or other arguments, to avoid shortfalls like reproducing common, but unjustified, opinions or biases, or being the victim of information manipulation.

As a third goal, I will address the challenge of intelligently selecting reliable information from the huge amount of heterogeneous resources available on the Web. The potential of answering queries by locating and combining information, which can be massively distributed across heterogeneous semantic resources, would allow to answer questions concerning e.g., the political trend of French boys and girls in the next elections, mining for instance real-time data from a semantic version of Twitter and from the websites of French political campaigns. Given their heterogeneity, data sources on the web can be noisy: they can provide information with different granularity and quality, require the integration of both structured and unstructured data (free text) for answer retrieval, or there can be either gaps in coverage or redundant data.

A step forward (the fourth goal) will concern dealing with the dynamic facet of the web, to allow the users to interact and dialogue with the debating platform interface (as a conversational agent), so that the system can provide an explanation and a justification for the results it provides, and the user can become aware of data provenance and trustworthiness. One of the main and complex issues to investigate concern finding strategies to handle the heterogeneous nature of the resources and ontologies, aggregating answers from different linked data resources, and carrying out reasoning on them. Moreover, existing work on (counter-)argument generation is limited to the reformulation of arguments mined from Wikipedia and few newspapers, and it is insufficient to generate effective and interactive explanatory arguments. Also investigating the automatic detection of the quality of an argument is an open challenge, to react e.g., in case of low quality arguments, through contextualized questions, including clarification questions (when the argument is not sufficiently elaborated), and requests for additional elaboration (when the argument lacks supporting evidence). Proactive behaviours should be managed during the argument-based dialogue (to mirror a crucial ability in human-human dialogues). Existing collaborations and connections with the Human Computer Interaction community will be strengthen to deal with the problems, desires, and requirements of technology users in the proposed application scenarios.

Such debating technologies will be applied in advanced decision support systems, a new generation of recommendation systems that engage with humans in a more sophisticated manner, and that are relevant for a wide spectrum of domains, including politics and legal (to facilitate active citizenship, e-governance and e-democracy), healthcare, education (for fact-checking and identifying of unjustified opinions or prejudices). Related to these aspects, I plan to keep on addressing research tasks going beyond binary abusive content classification, including identifying the target of abuse as well as automatically moderating content (through, for instance, counter-argumentation). Considerable challenges and unaddressed frontiers remain in the current systems and should be addressed, spanning technical, social and ethical dimensions. These issues constrain abusive content detection research, limiting its impact on the development of real-world detection systems.

Summary of the available resources created as contributions of my research:

- ALOOF Object and relations, Keyword Linking, Default Knowledge (`https://project.inria.fr/aloof/data/`)
- WASABI Song Corpus (`https://github.com/micbuffa/WasabiDataset`)
- AbstRCT - Argument Mining corpus on the health care domain (`https://gitlab.com/tomaye/abstrct/`
- Argument Mining corpus on the speeches and official declarations issued by Nixon and Kennedy (`https://dh.fbk.eu/resources/political-argumentation`)

- USElecDeb60To16 v.01 - Argument Mining corpus on the US Presidential Campaign Debates (`https://github.com/ElecDeb60To16/Dataset`)
- NoDE - A Benchmark of Natural Argumentation (`http://www-sop.inria.fr/NoDE/NoDE-xml.html`)
- DART - Dataset of Arguments and their Relations on Twitter (`https://www.aclweb.org/anthology/L16-1200.pdf`)

Bibliography

[1] Aseel Addawood and Masooda Bashir. "what is your evidence?" A study of controversial topics on social media. In *Proceedings of the Third Workshop on Argument Mining, hosted by the 54th Annual Meeting of the Association for Computational Linguistics, ArgMining@ACL 2016, August 12, Berlin, Germany*, 2016.

[2] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. Sentiment analysis of twitter data. In *Proceedings of the workshop on languages in social media*, pages 30–38. Association for Computational Linguistics, 2011.

[3] Sweta Agrawal and Amit Awekar. Deep learning for detecting cyberbullying across multiple social media platforms. In *ECIR*, pages 141–153, 2018.

[4] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proc. of COLING 2018*, pages 1638–1649, 2018.

[5] Khalid Al Khatib, Henning Wachsmuth, Johannes Kiesel, Matthias Hagen, and Benno Stein. A news editorial corpus for mining argumentation strategies. In *Proceedings of COLING 2016*, pages 3433–3443, 2016.

[6] Khalid Al Khatib, Henning Wachsmuth, Kevin Lang, Jakob Herpel, Matthias Hagen, and Benno Stein. Modeling deliberative argumentation strategies on wikipedia. In *Proceedings of ACL*, pages 2545–2555, 2018.

[7] Mehdi Allahyari, Seyed Amin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D. Trippe, Juan B. Gutierrez, and Krys Kochut. Text summarization techniques: A brief survey. *CoRR*, abs/1707.02268, 2017.

[8] Alo Allik, Florian Thalmann, and Mark Sandler. MusicLynx: Exploring music through artist similarity graphs. In *Companion Proc. (Dev. Track) The Web Conf. (WWW 2018)*, 2018.

[9] Omar Alonso and Kyle Shiells. Timelines as summaries of popular scheduled events. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 1037–1044. ACM, 2013.

[10] Leila Amgoud and Henri Prade. Can AI models capture natural language argumentation? *International Journal of Cognitive Informatics and Natural Intelligence*, 2013.

[11] Pramod Anantharam, Payam Barnaghi, Krishnaprasad Thirunarayan, and Amit Sheth. Extracting city traffic events from social streams. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(4):43, 2015.

[12] Reid Andersen, Fan Chung, and Kevin Lang. Local graph partitioning using pagerank vectors. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 475–486. IEEE, 2006.

[13] Ion Androutsopoulos and Prodromos Malakasiotis. A survey of paraphrasing and textual entailment methods. *J. Artif. Int. Res.*, 38(1):135–187, May 2010.

[14] Alessio Palmero Aprosio, Claudio Giuliano, and Alberto Lavelli. Automatic expansion of dbpedia exploiting wikipedia cross-language information. In *ESWC*, pages 397–411, 2013.

[15] Luis Enrique Argota Vega, Jorge Carlos Reyes-Magaña, Helena Gómez-Adorno, and Gemma Bel-Enguix. MineriaUNAM at SemEval-2019 task 5: Detecting hate speech in twitter using multiple features in a combinatorial framework. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 447–452, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[16] Rachit Arora and Balaraman Ravindran. Latent dirichlet allocation based multi-document summarization. In *Proceedings of the second workshop on Analytics for noisy unstructured text data*, pages 91–97. ACM, 2008.

[17] Pinar Arslan, Michele Corazza, Elena Cabrio, and Serena Villata. Overwhelmed by Negative Emotions? Maybe You Are Being Cyber-bullied! In *SAC 2019 - The 34th ACM/SIGAPP Symposium On Applied Computing*, Limassol, Cyprus, 2019.

[18] Kevin D. Ashley and Vern A. Walker. From information retrieval (IR) to argument retrieval (AR) for legal cases: Report on a baseline study. In *Legal Knowledge and Information Systems - JURIX*, pages 29–38, 2013.

[19] Farzindar Atefeh and Wael Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.

[20] Jack Atherton and Blair Kaneshiro. I said it first: Topological analysis of lyrical influence networks. In *ISMIR*, pages 654–660, 2016.

[21] Katie Atkinson, Pietro Baroni, Massimiliano Giacomin, Anthony Hunter, Henry Prakken, Chris Reed, Guillermo Ricardo Simari, Matthias Thimm, and Serena Villata. Towards artificial argumentation. *AI Magazine*, 38(3):25–36, 2017.

[22] Xiaoyu Bai, Flavio Merenda, Claudia Zaghi, Tommaso Caselli, and Malvina Nissim. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[23] Xiaoyu Bai, Flavio Merenda, Claudia Zaghi, Tommaso Caselli, and Malvina Nissim. Rug @ EVALITA 2018: Hate speech detection in italian social media. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*, 2018.

[24] Ivana Balazevic, Carl Allen, and Timothy Hospedales. TuckER: Tensor factorization for knowledge graph completion. In *Proc. of EMNLP-IJCNLP 2019*, pages 5185–5194, 2019.

[25] Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. Stance classification of context-dependent claims. In *Proceedings of EACL*, pages 251–261, 2017.

[26] Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. Stance classification of context-dependent claims. In *Proc. of EACL 2017*, pages 251–261, 2017.

[27] Adriano Baratè, Luca A. Ludovico, and Enrica Santucci. A semantics-driven approach to lyrics segmentation. In *2013 8th International Workshop on Semantic and Social Media Adaptation and Personalization*, pages 73–79, Dec 2013.

[28] Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. What does this emoji mean? a vector space skip-gram model for twitter emojis. In *Language Resources and Evaluation conference, LREC*, Portoroz, Slovenia, May 2016.

[29] Ken Barker, Bhalchandra Agashe, Shaw Yi Chaw, James Fan, Noah S. Friedland, Michael Robert Glass, Jerry R. Hobbs, Eduard H. Hovy, David J. Israel, Doo Soon Kim, Rutu Mulkar-Mehta, Sourabh Patwardhan, Bruce W. Porter, Dan Tecuci, and Peter Z. Yeh. Learning by reading: A prototype system, performance baseline and lessons learned. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 280–286, 2007.

[30] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 238–247, June 2014. DOI: `10.3115/v1/P14-1023`.

[31] Marco Baroni and Roberto Zamparelli. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1183–1193, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[32] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *Knowledge Eng. Review*, 26(4):365–410, 2011.

[33] Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606, 2016.

[34] Lawrence W. Barsalou. Simulation, situated conceptualization, and prediction. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521):1281–1289, 2009. DOI: `10.1098/rstb.2008.0319`.

[35] Mark A. Bartsch and Gregory H. Wakefield. Audio thumbnailing of popular music using chroma-based representations. *Trans. Multi.*, 7(1):96–104, February 2005.

[36] Regina Barzilay and Kathleen R. McKeown. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–327, 2005.

[37] Amparo Elizabeth Cano Basave and Yulan He. A study of the impact of persuasive argumentation in political debates. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1405–1413, 2016.

[38] Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of SEMEVAL 2019*, Minneapolis, Minnesota, USA, June 2019.

[39] Valerio Basile, Soufian Jebbara, Elena Cabrio, and Philipp Cimiano. Populating a Knowledge Base with Object-Location Relations Using Distributional Semantics. In *20th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2016)*, pages 34 – 50, Bologna, Italy, November 2016. DOI: 10.1007/978-3-319-49004-5_3.

[40] Valerio Basile and Malvina Nissim. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta, 2013.

[41] Elisa Bassignana, Valerio Basile, and Viviana Patti. Hurtlex: A multilingual lexicon of words to hurt. In *5th Italian Conference on Computational Linguistics, CLiC-it 2018*, volume 2253, pages 1–6. CEUR-WS, 2018.

[42] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[43] Michael Beetz, Ferenc Bálint-Benczédi, Nico Blodow, Daniel Nyga, Thiemo Wiedemeyer, and Zoltán-Csaba Marton. Robosherlock: Unstructured information processing for robot perception. In *ICRA*, 2015.

[44] Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: A pretrained language model for scientific text. In *Proc. of EMNLP-IJCNLP 2019*, pages 3615–3620, 2019.

[45] Trevor J. M. Bench-Capon, D. Lowes, and A. M. McEnery. Argument-based explanation of logic programs. *Knowl.-Based Syst.*, 4(3):177–183, 1991.

[46] Trevor J. M. Bench-Capon and Giovanni Sartor. Theory based explanation of case law domains. In *ICAIL*, pages 12–21, 2001.

[47] Mohamed S. Benlamine, Serena Villata, Ramla Ghali, Claude Frasson, Fabien L. Gandon, and Elena Cabrio. Persuasive argumentation and emotions: An empirical evaluation with users. In *Human-Computer Interaction. User Interface Design, Development and Multimodality - 19th International Conference, HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings, Part I*, pages 659–671, 2017.

[48] Sahbi Benlamine, Maher Chaouachi, Serena Villata, Elena Cabrio, Claude Frasson, and Fabien Gandon. Emotions in argumentation: an empirical evaluation. In Qiang Yang and Michael Wooldridge, editors, *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, pages 156–163. AAAI Press, 2015.

[49] Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 481–490, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[50] Linn Bergelid. Classification of explicit music content using lyrics and music metadata, 2018.

[51] Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[52] Floris Bex and Douglas Walton. Burdens and standards of proof for inference to the best explanation. In Radboud Winkels, editor, *Legal Knowledge and Information Systems - JURIX 2010: The Twenty-Third Annual Conference on Legal Knowledge and Information Systems, Liverpool, UK, 16-17 December 2010*, volume 223 of *Frontiers in Artificial Intelligence and Applications*, pages 37–46. IOS Press, 2010.

[53] Shraey Bhatia, Jey Han Lau, and Timothy Baldwin. Automatic labelling of topics with neural embeddings. *arXiv preprint arXiv:1612.05340*, 2016.

[54] Christian Bizer et al. DBpedia - a crystallization point for the web of data. *Web Semant.*, 7(3):154–165, 2009.

[55] P. Blackburn, J. Bos, M. Kohlhase, and H. de Nivelle. Inference and computational semantics. *Studies in Linguistics and Philosophy, Computing Meaning*, 77(2):11–28, 2001.

[56] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[57] Sebastian Blohm and Philipp Cimiano. Using the web to reduce data sparseness in pattern-based information extraction. In *Knowledge Discovery in Databases: PKDD 2007, 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings*, pages 18–29, 2007. DOI: 10.1007/978-3-540-74976-9_6.

[58] Sebastian Blohm, Philipp Cimiano, and Egon Stemle. Harvesting relations from the web -quantifiying the impact of filtering functions. In *Proceedings of the 22nd Conference on Artificial Intelligence (AAAI-07)*, pages 1316–1323. Association for the Advancement of Artificial Intelligence (AAAI), Juli 2007.

[59] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[60] Joshua E. Blumenstock. Size matters: Word count as a measure of quality on wikipedia. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 1095–1096, New York, NY, USA, 2008. ACM.

[61] Guido Boella, Dov M. Gabbay, Leendert W. N. van der Torre, and Serena Villata. Support in abstract argumentation. In Pietro Baroni, Federico Cerutti, Massimiliano Giacomin, and Guillermo Ricardo Simari, editors, *COMMA*, volume 216 of *Frontiers in Artificial Intelligence and Applications*, pages 111–122. IOS Press, 2010.

[62] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[63] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA, 2008. ACM. DOI: 10.1145/1376616.1376746.

[64] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. In *Experimental Robotics - The 13th International Symposium on Experimental Robotics, ISER 2012, June 18-21, 2012, Québec City, Canada*, pages 481–495. 2012. DOI: 10.1007/978-3-319-00065-7_33.

[65] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc., 2013.

[66] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proc. of NIPS 2013*, pages 2787–2795, 2013.

[67] Johan Bos and Katja Markert. When logical inference helps determining textual entailment (and when it doesn't). In *Proc. of the 2nd PASCAL Workshop on Recognizing Textual Entailment*, October 2006.

[68] Tom Bosc, Elena Cabrio, and Serena Villata. DART: a dataset of arguments and their relations on twitter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*, 2016.

[69] Tom Bosc, Elena Cabrio, and Serena Villata. Tweeties squabbling: Positive and negative results in applying argument mining on social media. In Pietro Baroni, Thomas F. Gordon, Tatjana Scheffler, and Manfred Stede, editors, *Computational Models of Argument - Proceedings of COMMA 2016, Potsdam, Germany, 12-16 September, 2016*, volume 287 of *Frontiers in Artificial Intelligence and Applications*, pages 21–32. IOS Press, 2016.

[70] Cristina Bosco, Felice Dell'Orletta, Fabio Poletto, Manuela Sanguinetti, and Maurizio Tesconi. Overview of the EVALITA 2018 hate speech detection task. In *Proceedings of EVALITA 2018*, 2018.

[71] David Brackett. *Interpreting Popular Music*. Cambridge University Press, 1995.

[72] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[73] Volha Bryl and Christian Bizer. Learning conflict resolution strategies for cross-language wikipedia data fusion. In *WWW (Companion Volume)*, pages 1129–1134, 2014.

[74] Katarzyna Budzynska, Mathilde Janier, Juyeon Kang, Chris Reed, Patrick Saint-Dizier, Manfred Stede, and Olena Yaskorska. Towards argument mining from dialogue. In *Proceedings of COMMA*, pages 185–196, 2014.

[75] Katarzyna Budzynska and Chris Reed. Whence inference. Technical report, University of Dundee, 2011.

[76] Michel Buffa and Jerome Lebrun. Real time tube guitar amplifier simulation using webaudio. In *Proc. 3rd Web Audio Conference (WAC 2017)*, 2017.

[77] Michel Buffa and Jerome Lebrun. Web audio guitar tube amplifier vs native simulations. In *Proc. 3rd Web Audio Conf. (WAC 2017)*, 2017.

[78] Michel Buffa, Jerome Lebrun, Jari Kleimola, Stéphane Letz, et al. Towards an open web audio plugin standard. In *Companion Proceedings of the The Web Conference 2018*, pages 759–766. International World Wide Web Conferences Steering Committee, 2018.

[79] Michel Buffa, Jerome Lebrun, Johan Pauwels, and Guillaume Pellerin. A 2 Million Commercial Song Interactive Navigator. In *WAC 2019 - 5th WebAudio Conference 2019*, Trondheim, Norway, December 2019.

[80] Michel Buffa, Jerome Lebrun, Guillaume Pellerin, and Stéphane Letz. Webaudio plugins in daws and for live performance. In *14th International Symposium on Computer Music Multidisciplinary Research (CMMR'19)*, 2019.

[81] Razvan C. Bunescu and Raymond J. Mooney. Subsequence kernels for relation extraction. In *Proceedings of the 18th International Conference on Neural Information Processing Systems*, NIPS'05, pages 171–178, Cambridge, MA, USA, 2005. MIT Press.

[82] Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, and Fabien Gandon. Natural language interaction with the web of data by mining its textual side. *Intelligenza Artificiale*, 6(2):121–133, 2012.

[83] Elena Cabrio, Julien Cojan, Alessio Palmero Aprosio, Bernardo Magnini, Alberto Lavelli, and Fabien Gandon. Qakis: an open domain QA system based on relational patterns. In Birte Glimm and David Huynh, editors, *Proceedings of the ISWC 2012 Posters & Demonstrations Track, Boston, USA, November 11-15, 2012*, volume 914 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2012.

[84] Elena Cabrio, Julien Cojan, and Fabien Gandon. Mind the Cultural Gap: Bridging Language-Specific DBpedia Chapters for Question Answering. In Paul Buitelaar and Philipp Cimiano, editors, *Towards the Multilingual Semantic Web*. Springer-Verlag Berlin Heidelberg, 2014.

[85] Elena Cabrio, Julien Cojan, Fabien Gandon, and Amine Hallili. Querying multilingual dbpedia with qakis. In *The Semantic Web: ESWC 2013 Satellite Events - ESWC 2013 Satellite Events, Montpellier, France, May 26-30, 2013, Revised Selected Papers*, pages 194–198, 2013.

[86] Elena Cabrio, Julien Cojan, Serena Villata, and Fabien Gandon. Argumentation-based inconsistencies detection for question-answering over dbpedia. In *NLP-DBPEDIA@ISWC*, 2013.

[87] Elena Cabrio, Alessio Palmero Aprosio, and Serena Villata. Reconciling Information in DBpedia through a Question Answering System. ISWC 2014 : 13th International Semantic Web Conference, October 2014. Poster.

[88] Elena Cabrio, Vivek Sachidananda, and Raphaël Troncy. Boosting qakis with multimedia answer visualization. In *The Semantic Web: ESWC 2014 Satellite Events - ESWC 2014 Satellite Events, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 298–303, 2014.

[89] Elena Cabrio and Serena Villata. Combining textual entailment and argumentation theory for supporting online debates interactions. In *Proceedings of ACL*, pages 208–212, 2012.

[90] Elena Cabrio and Serena Villata. Natural language arguments: A combined approach. In *ECAI 2012 - 20th European Conference on Artificial Intelligence. Including Prestigious Applications of Artificial Intelligence (PAIS-2012) System Demonstrations Track, Montpellier, France, August 27-31 , 2012*, pages 205–210, 2012.

[91] Elena Cabrio and Serena Villata. A natural language bipolar argumentation approach to support users in online debate interactions†. *Argument & Computation*, 4(3):209–230, 2013.

[92] Elena Cabrio and Serena Villata. Five years of argument mining: a data-driven analysis. In Jérôme Lang, editor, *Proceedings of IJCAI*, pages 5427–5433. ijcai.org, 2018.

[93] Elena Cabrio, Serena Villata, and Alessio Palmero Aprosio. A RADAR for information reconciliation in question answering systems over linked data. *Semantic Web*, 8(4):601–617, 2017.

[94] Elena Cabrio, Serena Villata, and Fabien Gandon. A support framework for argumentative discussions management in the web. In *The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26-30, 2013. Proceedings*, pages 412–426, 2013.

[95] Elena Cabrio, Serena Villata, and Fabien Gandon. Classifying inconsistencies in dbpedia language specific chapters. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014*, pages 1443–1450, 2014.

[96] José Camacho-Collados, Mohammad Taher Pilehvar, and Roberto Navigli. NASARI: A novel approach to a semantically-aware representation of items. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 567–577, 2015. DOI: `10.3115/v1/N15-1059`.

[97] Erion Çano and Maurizio Morisio. Music mood dataset creation based on last.fm tags. In *2017 International Conference on Artificial Intelligence and Applications, Vienna Austria*, May 2017.

[98] Jean Carletta. Assessing agreement on classification tasks: the kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[99] Miguel Ángel Álvarez Carmona, Estefanía Guzmán-Falcón, Manuel Montes-y-Gómez, Hugo Jair Escalante, Luis Villaseñor Pineda, Verónica Reyes-Meza, and Antonio Rico Sulayes. Overview of MEX-A3T at IberEval 2018: Authorship and Aggressiveness Analysis in Mexican Spanish Tweets. In *Proceedings of IberEval 2018*, pages 74–96, 2018.

[100] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. On the acceptability of arguments in bipolar argumentation frameworks. In *Proc. of the 8th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), LNCS 3571*, pages 378–389, July 2005.

[101] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Coalitions of arguments: A tool for handling bipolar argumentation frameworks. *Int. J. Intell. Syst.*, 25(1):83–109, 2010.

[102] Claudette Cayrol and Marie-Christine Lagasquie-Schiex. Bipolarity in argumentation graphs: Towards a better understanding. In *Procs of SUM 2011*, volume 6929 of *LNCS*, pages 137–148. Springer, 2011.

[103] Wei Chai and Barry Vercoe. Music thumbnailing via structural analysis. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 223–226, 01 2003.

[104] Chih-Chung Chang and Chih-Jen Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.

[105] Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi, and Puneet Agrawal. Semeval-2019 task 3: Emocontext contextual emotion detection in text. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 39–48, 2019.

[106] Heng-Tze Cheng, Yi-Hsuan Yang, Yu-Ching Lin, and Homer H. Chen. Multimodal structure segmentation and analysis of music using audio and textual information. In *2009 IEEE International Symposium on Circuits and Systems*, pages 1677–1680, May 2009.

[107] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494. Association for Computational Linguistics, 2016.

[108] Carlos Iván Chesñevar, Jarred McGinnis, Sanjay Modgil, Iyad Rahwan, Chris Reed, Guillermo Ricardo Simari, Matthew South, Gerard Vreeswijk, and Steven Willmott. Towards an argument interchange format. *Knowledge Eng. Review*, 21(4):293–316, 2006.

[109] Gennaro Chierchia and Sally McConnell-Ginet. *Meaning and Grammar: An Introduction to Semantics 2nd ed.* Cambridge, MA: MIT Press, 2000.

[110] Hyojin Chin, Jayong Kim, Yoonjong Kim, Jinseop Shin, and Mun Y Yi. Explicit content detection in music lyrics using machine learning. In *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 517–521. IEEE, 2018.

[111] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, 2014.

[112] François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[113] Mark Cieliebak, Jan Milan Deriu, Dominic Egger, and Fatih Uzdilli. A twitter corpus and benchmark resources for german sentiment analysis. In *5th International Workshop on Natural Language Processing for Social Media, Boston, MA, USA*, pages 45–51. Association for Computational Linguistics, 2017.

[114] Philipp Cimiano, Vanessa Lopez, Christina Unger, Elena Cabrio, Axel-Cyrille Ngonga Ngomo, and Sebastian Walter. Multilingual question answering over linked data (qald-3): Lab overview. In Pamela Forner, Henning Müller, Roberto Paredes, Paolo Rosso, and Benno Stein, editors, *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, pages 321–332, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[115] Philipp Cimiano and Johanna Wenderoth. Automatically Learning Qualia Structures from the Web. In Timothy Baldwin, Anna Korhonen, and Aline Villavicencio, editors, *Proceedings of the ACL Workshop on Deep Lexical Acquisition*, pages 28–37. Association for Computational Linguistics, 2005. DOI: `10.3115/1631850.1631854`.

[116] Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. Multi-task learning in deep neural networks at EVALITA 2018. In *EVALITA 2018*, 2018.

[117] Alina Maria Ciobanu and Anca Dinu. Alternative measures of word relatedness in distributional semantics. In *Joint Symposium on Semantic Processing*, page 80, 2013.

[118] Alice Cohen-Hadria and Geoffroy Peeters. Music Structure Boundaries Estimation Using Multiple Self-Similarity Matrices as Input Depth of Convolutional Neural Networks. In *AES International Conference Semantic Audio 2017*, Erlangen, Germany, June 2017.

[119] Julien Cojan, Elena Cabrio, and Fabien Gandon. Filling the gaps among dbpedia multilingual chapters for question answering. In *Web Science 2013 (co-located with ECRC), WebSci '13, Paris, France, May 2-4, 2013*, pages 33–42, 2013.

[120] Stephen Coleman, Jay G. Blumler, Giles Moss, and Matt Homer. The 2015 televised election debates; democracy on demand? *Leeds University Press*, 12 2015.

[121] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. NLP (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

[122] Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata. Comparing different supervised approaches to hate speech detection. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*, 2018.

[123] Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata. InriaFBK at Germeval 2018: Identifying Offensive Tweets Using Recurrent Neural Networks. In *GermEval 2018 Workshop*, Vienna, Austria, September 2018.

[124] Michele Corazza, Stefano Menini, Pinar Arslan, Rachele Sprugnoli, Elena Cabrio, Sara Tonelli, and Serena Villata. Inriafbk at germeval 2018: Identifying offensive tweets using recurrent neural networks. In *GermEval 2018 Workshop*, 2018.

[125] Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. Cross-platform evaluation for italian hate speech detection. In *Proceedings of the 6th Italian Conference on Computational Linguistics*, 2019.

[126] Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. Inriafbk drawing attention to offensive language at germeval2019. In *Proceedings of the 15th Conference on Natural Language Processing, KONVENS 2019, Erlangen, Germany, October 9-11, 2019*, 2019.

[127] Michele Corazza, Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. A multilingual evaluation for online hate speech detection. *ACM Trans. Internet Techn.*, 20(2):10:1–10:22, 2020.

[128] International Press Telecommunications Council. SportsML: A solution for sharing sports data. `https://iptc.org/standards/sportsml-g2/`, 2017. [Accessed 03-01-2017].

[129] Robert Craven, Francesca Toni, Cristian Cadar, Adrian Hadad, and Matthew Williams. Efficient argumentation for medical decision-making. In *Proc. of KR 2012*, pages 598–602, 2012.

[130] Hamish Cunningham, Diana Maynard, Kalina Bontcheva, and Valentin Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, 2002.

[131] Célia da Costa Pereira, Andrea Tettamanzi, and Serena Villata. Changing one's mind: Erase or rewind? In *Procs of IJCAI 2011*, pages 164–171. IJCAI/AAAI, 2011.

[132] I. Dagan, B. Dolan, B. Magnini, and D. Roth. Recognizing textual entailment: Rational, evaluation and approaches. *JNLE*, 15(04):i–xvii, 2009.

[133] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 121–124, New York, NY, USA, 2013. ACM. DOI: 10.1145/2506182.2506198.

[134] Danica Damljanovic, Milan Agatonovic, and Hamish Cunningham. Freya: an interactive way of querying linked data using natural language. In *Procs of ESWC 2012*, pages 125–138. Springer, 2012.

[135] Thomas Davidson, Dana Warmsley, Michael W. Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017.*, pages 512–515, 2017.

[136] Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, pages 273–274, Republic and Canton of Geneva, Switzerland, 2016. International World Wide Web Conferences Steering Committee.

[137] Steven B. Davis and Paul Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *ACOUSTICS, SPEECH AND SIGNAL PROCESSING, IEEE TRANSACTIONS ON*, pages 357–366, 1980.

[138] Tim Van de Cruys, Thierry Poibeau, and Anna Korhonen. A tensor-based factorization model of semantic compositionality. In *Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies (NAACL-HLT 2013)*, pages 1142–1151, Atlanta, GA, US, 2013. Association for Computational Linguistics (ACL).

[139] Rémi Delbouys, Romain Hennequin, Francesco Piccoli, Jimena Royo-Letelier, and Manuel Moussallam. Music mood detection based on audio and lyrics with deep neural net. *arXiv preprint arXiv:1809.07276*, 2018.

[140] Jean Yves Delort, Bernadette Bouchon-Meunier, and Maria Rifqi. Enhanced web document summarization using hyperlinks. In *Proceedings of the Fourteenth ACM Conference on Hypertext and Hypermedia*, HYPERTEXT '03, pages 208–215, New York, NY, USA, 2003. ACM.

[141] Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49, 2015.

[142] Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proc. of AAAI 2018*, pages 1811–1818, February 2018.

[143] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[144] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proc. of NAACL-HLT 2019*, pages 4171–4186, 2019.

[145] Lee R. Dice. Measures of the amount of ecologic association between species. *Journal of Ecology*, 26:297–302, 1945.

[146] Karthik Dinakar, Birago Jones, Catherine Havasi, Henry Lieberman, and Rosalind W. Picard. Common Sense Reasoning for Detection, Prevention, and Mitigation of Cyberbullying. *TiiS*, 2(3):18:1–18:30, 2012.

[147] Paul Doran, Valentina Tamma, Ignazio Palmisano, and Terry Payne. Efficient argumentation over ontology correspondences. In *Procs of AAMAS 2009*, pages 1241–1242, 2009.

[148] Cássia Trojahn dos Santos and Jérôme Euzenat. Consistency-driven argumentation for alignment agreement. In *Procs of OM 2010, CEUR Workshop Proceedings 689*, 2010.

[149] Cícero Nogueira dos Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1818–1826, 2014.

[150] Wenwen Dou, K Wang, William Ribarsky, and Michelle Zhou. Event detection in social media data. In *IEEE VisWeek Workshop on Interactive Visual Text Analytics-Task Driven Analytics of Social Media Content*, pages 971–980, 2012.

[151] Phan M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[152] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2):321–358, 1995.

[153] Paul E. Dunne, Anthony Hunter, Peter McBurney, Simon Parsons, and Michael Wooldridge. Weighted argument systems: Basic definitions, algorithms, and complexity results. *Artif. Intell.*, 175(2):457–486, 2011.

[154] Mihai Dusmanu, Elena Cabrio, and Serena Villata. Argument mining on twitter: Arguments, facts and sources. In *Proceedings of EMNLP*, pages 2317–2322, 2017.

[155] Rory Duthie and Katarzyna Budzynska. A deep modular RNN approach for ethos mining. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, pages 4041–4047, 2018.

[156] Rory Duthie, Katarzyna Budzynska, and Chris Reed. Mining ethos in political debate. In *COMMA*, pages 299–310, 2016.

[157] Amosse Edouard, Elena Cabrio, Sara Tonelli, and Nhan Le-Thanh. Graph-based event extraction from twitter. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 222–230, Varna, Bulgaria, September 2017. INCOMA Ltd.

[158] Amosse Edouard, Elena Cabrio, Sara Tonelli, and Nhan Le Thanh. Semantic Linking for Event-Based Classification of Tweets. In *Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics (CICLing)*, 2017.

[159] Amosse Edouard, Elena Cabrio, Sara Tonelli, and Nhan Le-Thanh. You'll never tweet alone: Building sports match timelines from microblog posts. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 214–221, Varna, Bulgaria, September 2017. INCOMA Ltd.

[160] Charlie Egan, Advaith Siddharthan, and Adam Z. Wyner. Summarising the points made in online political debates. In *Proceedings of the Third Workshop on Argument Mining*, 2016.

[161] Steffen Eger, Johannes Daxenberger, and Iryna Gurevych. Neural end-to-end learning for computational argumentation mining. In *Proc. of ACL 2017*, pages 11–22, 2017.

[162] Roxanne El Baff, Henning Wachsmuth, Khalid Al-Khatib, and Benno Stein. Challenge or empower: Revisiting argumentation quality in a news editorial corpus. In *Proceedings of the 22nd Conference on Computational Natural Language Learning (CoNLL 2018)*, pages 454–464. Association for Computational Linguistics, 2018.

[163] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

[164] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[165] Oren Etzioni. Machine reading at web scale. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 2–2. ACM, 2008. DOI: `10.1145/1341531.1341533`.

[166] Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. Open information extraction: The second generation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume One*, IJCAI'11, pages 3–10. AAAI Press, 2011. DOI: `10.5591/978-1-57735-516-8/IJCAI11-012`.

[167] Thomas Faeulhammer, Rares Ambrus, Chris Burbridge, Michael Zillich, John Folkesson, Nick Hawes, Patric Jensfelt, and Markus Vincze. Autonomous learning of object models on a mobile robot. *IEEE RAL*, PP(99):1–1, 2016.

[168] Mohamed Abdel Fattah. A hybrid machine learning model for multi-document summarization. *Applied intelligence*, 40(4):592–600, 2014.

[169] Mohamed Abdel Fattah and Fuji Ren. Ga, mr, ffnn, pnn and gmm based models for automatic text summarization. *Comput. Speech Lang.*, 23(1):126–144, January 2009.

[170] Michael Fell. Lyrics classification. Master's thesis, Saarland University, Germany, 2014.

[171] Michael Fell, Elena Cabrio, Michele Corazza, and Fabien Gandon. Comparing Automated Methods to Detect Explicit Content in Song Lyrics. In *Conference on Recent Advances in Natural Language Processing (RANLP)*, Varna, Bulgaria, September 2019.

[172] Michael Fell, Elena Cabrio, Fabien Gandon, and Alain Giboin. Song lyrics summarization inspired by audio thumbnailing. In *Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP)*, 2019.

[173] Michael Fell, Elena Cabrio, Elmahdi Korfed, Michel Buffa, and Fabien Gandon. Love me, love me, say (and write!) that you love me: Enriching the WASABI song corpus with lyrics annotations. In *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020*, pages 2138–2147, 2020.

[174] Michael Fell, Yaroslav Nechaev, Gabriel Meseguer Brocal, Elena Cabrio, Fabien Gandon, and Geoffroy Peeters. Lyrics segmentation via bimodal text-audio representation. *Journal of Natural Language Engineering (to appear)*, 2020.

[175] Michael Fell, Yaroslav Nechaev, Elena Cabrio, and Fabien Gandon. Lyrics segmentation: Textual macrostructure detection using convolutions. In Emily M. Bender, Leon Derczynski, and Pierre Isabelle, editors, *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018*, pages 2044–2054. Association for Computational Linguistics, 2018.

[176] Michael Fell and Caroline Sporleder. Lyrics-based analysis and classification of music. In Jan Hajic and Junichi Tsujii, editors, *COLING*, pages 620–631. ACL, 2014.

[177] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.

[178] Vanessa Wei Feng and Graeme Hirst. Classifying arguments by scheme. In *Proceedings of ACL*, pages 987–996, 2011.

[179] Elisabetta Fersini, Paolo Rosso, and Maria Anzovino. Overview of the task on automatic misogyny identification at ibereval 2018. In *IberEval@SEPLN*, volume 2150 of *CEUR Workshop Proceedings*, pages 214–228. CEUR-WS.org, 2018.

[180] Thomas Fillon, Joséphine Simonnot, Marie-France Mifune, Stéphanie Khoury, Guillaume Pellerin, and Maxime Le Coz. Telemeta: An open-source web framework for ethnomusicological audio archives management and automatic analysis. In *Proceedings of the 1st International Workshop on Digital Libraries for Musicology*, pages 1–8. ACM, 2014.

[181] Ross Finman, Thomas Whelan, Michael Kaess, and John J Leonard. Toward lifelong object segmentation from change detection in dense rgb-d maps. In *ECMR*. IEEE, 2013.

[182] Darja Fišer, Ruihong Huang, Vinodkumar Prabhakaran, Rob Voigt, Zeerak Waseem, and Jacqueline Wernimont. Proceedings of the 2nd workshop on abusive language online (alw2). In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*. Association for Computational Linguistics, 2018.

[183] Jonathan Foote. Automatic audio segmentation using a measure of audio novelty. In *Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference on*, volume 1, pages 452–455. IEEE, 2000.

[184] Paula Fortuna, Ilaria Bonavita, and Sérgio Nunes. Merging datasets for hate speech classification in italian. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy, December 12-13, 2018.*, 2018.

[185] Antigoni-Maria Founta, Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Athena Vakali, and Ilias Leontiadis. A unified deep learning architecture for abuse detection. *CoRR*, abs/1802.00385, 2018.

[186] Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. Large scale crowdsourcing and characterization of twitter abusive behavior. In *ICWSM*, pages 491–500, 2018.

[187] Lutz Frommberger and Diedrich Wolter. Structural knowledge transfer by spatial abstraction for reinforcement learning agents. *Adaptive Behavior*, 18(6):507–525, December 2010.

[188] Takuya Fujishima. Realtime chord recognition of musical sound: a system using common lisp music. In *ICMC*. Michigan Publishing, 1999.

[189] Andrea Galassi, Marco Lippi, and Paolo Torroni. Argumentative link prediction using residual networks and multi-objective learning. In *Proc. of ArgMining 2018 workshop*, pages 1–10, 2018.

[190] Björn Gambäck and Utpal Kumar Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90. Association for Computational Linguistics, 2017.

[191] Guglielmo Gemignani, Roberto Capobianco, Emanuele Bastianelli, Domenico Bloisi, Luca Iocchi, and Daniele Nardi. Living with robots: Interactive environmental knowledge acquisition. *Robotics and Autonomous Systems*, 2016.

[192] Daniel Gerber and Axel-Cyrille Ngonga Ngomo. Bootstrapping the linked data web. In *1st Workshop on Web Scale Knowledge Extraction @ ISWC 2011*, 2011.

[193] Chris Gibbs and Richard Haynes. A phenomenological investigation into how twitter has changed the nature of sport media relations. *International Journal of Sport Communication*, 6(4):394–408, 2013.

[194] Roxana Girju, Adriana Badulescu, and Dan Moldovan. Learning semantic constraints for the automatic discovery of part-whole relations. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 1–8, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. DOI: `10.3115/1073445.1073456`.

[195] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[196] Thomas F. Gordon, Henry Prakken, and Douglas Walton. The carneades model of argument and burden of proof. *Artif. Intell.*, 171(10-15):875–896, 2007.

[197] Theodosios Goudas, Christos Louizos, Georgios Petasis, and Vangelis Karkaletsis. Argument extraction from news, blogs, and the social web. *International Journal on Artificial Intelligence Tools*, 24(5), 2015.

[198] Mario Graff, Sabino Miranda-Jiménez, Eric Sadit Tellez, Daniela Moctezuma, Vladimir Salgado, José Ortiz-Bejar, and Claudia N. Sánchez. INGEOTEC at MEX-A3T: author profiling and aggressiveness analysis in twitter using μtc and evomsa. In *IberEval 2018*, pages 128–133, 2018.

[199] Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomas Mikolov. Learning word vectors for 157 languages. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[200] Nancy Green. Argumentation for scientific claims in a biomedical research article. In *Proc. of ArgNLP 2014 workshop*, 2014.

[201] Nancy Green. Annotating evidence-based argumentation in biomedical text. *IEEE BIBM 2015*, pages 922–929, 2015.

[202] Nancy Green. Towards mining scientific discourse using argumentation schemes. *Argument and Computation*, 9:121–135, 2018.

[203] Kathrin Grosse, María Paula González, Carlos Iván Chesñevar, and Ana Gabriela Maguitman. Integrating argumentation and sentiment analysis for mining opinions from twitter. *AI Commun.*, 28(3):387–401, 2015.

[204] Ivan Habernal and Iryna Gurevych. Exploiting debate portals for semi-supervised argumentation mining in user-generated web discourse. In *Proceedings of EMNLP*, pages 2127–2137, 2015.

[205] Ivan Habernal and Iryna Gurevych. Which argument is more convincing? Analyzing and predicting convincingness of web arguments using bidirectional LSTM. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1589–1599. Association for Computational Linguistics, 2016.

[206] Ivan Habernal and Iryna Gurevych. Argumentation mining in user-generated web discourse. *Comput. Linguist.*, 43(1):125–179, 2017.

[207] Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. Before name-calling: Dynamics and triggers of ad hominem fallacies in web argumentation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 386–396, 2018.

[208] Shohreh Haddadan, Elena Cabrio, and Serena Villata. Disputool - A tool for the argumentative analysis of political debates. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6524–6526, 2019.

[209] Shohreh Haddadan, Elena Cabrio, and Serena Villata. Yes, we can! mining arguments in 50 years of US presidential campaign debates. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers*, pages 4684–4690, 2019.

[210] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370. Association for Computational Linguistics, 2009.

[211] Marc Hanheide, Charles Gretton, Richard Dearden, Nick Hawes, Jeremy L. Wyatt, Andrzej Pronobis, Alper Aydemir, Moritz Göbelbecker, and Hendrik Zender. Exploiting probabilistic knowledge under uncertain sensing for efficient robot behaviour. In *IJCAI'11*, Barcelona, Spain, July 2011.

[212] John Hannon, Kevin McCarthy, James Lynch, and Barry Smyth. Personalized and automatic social summarization of events in video. In *Proceedings of the 16th international conference on Intelligent user interfaces*, pages 335–338. ACM, 2011.

[213] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335 – 346, 1990. DOI: 10.1016/0167-2789(90)90087-6.

[214] Stevan Harnad. Categorical perception. In L. Nadel, editor, *Encyclopedia of Cognitive Science*, pages 67–4. Nature Publishing Group, 2003.

[215] Stevan Harnad. To cognize is to categorize: Cognition is categorization. *Handbook of categorization in cognitive science*, pages 20–45, 2005.

[216] Zellig Harris. Distributional structure. *Word*, 10(23):146–162, 1954. DOI: 10.1080/00437956.1954.11659520.

[217] Mahmud Hasan, Mehmet A Orgun, and Rolf Schwitter. Twitternews: real time event detection from the twitter data stream. *PeerJ PrePrints*, 4:e2297v1, 2016.

[218] Ruifang He and Xingyi Duan. Twitter summarization based on social network and sparse reconstruction. In *AAAI*, 2018.

[219] Benjamin Heinzerling and Michael Strube. Bpemb: Tokenization-free pre-trained subword embeddings in 275 languages. In *Proc. of LREC 2018*, pages 2989–2993, 2018.

[220] Romain Hennequin, Anis Khlif, Felix Voituret, and Manuel Moussallam. Spleeter: A fast and state-of-the art music source separation tool with pre-trained models. Late-Breaking/Demo ISMIR 2019, November 2019. Deezer Research.

[221] Leonhard Hennig. Topic-based multi-document summarization with probabilistic latent semantic analysis. In *Proceedings of the International Conference RANLP-2009*, pages 144–149, 2009.

[222] Stella Heras, Katie Atkinson, Vicente J. Botti, Floriana Grasso, Vicente Julián, and Peter McBurney. How argumentation can enhance dialogues in social networks. In *Computational Model of Arguments (COMMA)*, pages 267–274, 2010.

[223] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *IEEE ICCV*, 2011.

[224] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[225] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 541–550. Association for Computational Linguistics, 2011.

[226] Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. Learning 5000 relational extractors. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 286–295, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[227] Richang Hong, Meng Wang, Guangda Li, Liqiang Nie, Zheng-Jun Zha, and Tat-Seng Chua. Multimedia question answering. *IEEE MultiMedia*, 19(4):72–78, 2012.

[228] Matthew Honnibal and Ines Montani. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*, 2017.

[229] Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, and Shivakant Mishra. Analyzing labeled cyberbullying incidents on the instagram social network. In *Social Informatics - 7th International Conference, SocInfo 2015, Beijing, China, December 9-12, 2015, Proceedings*, pages 49–66, 2015.

[230] Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard O. Han, Qin Lv, and Shivakant Mishra. Prediction of Cyberbullying Incidents on the Instagram Social Network. *CoRR*, abs/1503.03909, 2015.

[231] Meishan Hu, Aixin Sun, Ee-Peng Lim, and Ee-Peng Lim. Comments-oriented blog summarization by sentence extraction. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, pages 901–904, New York, NY, USA, 2007. ACM.

[232] Tianran Hu, Han Guo, Hao Sun, Thuy-vy Thi Nguyen, and Jiebo Luo. Spice up your chat: The intentions and sentiment effects of using emojis. In *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada*, pages 102–111, 2017.

[233] Xiao Hu, J Stephen Downie, and Andreas F Ehmann. Lyric text mining in music mood classification. *American music*, 183(5,049):2–209, 2009.

[234] Yajie Hu, Xiaoou Chen, and Deshun Yang. Lyric-based song emotion detection with affective lexicon and fuzzy clustering method. In *ISMIR*, 2009.

[235] Xinyu Hua, Mitko Nikolov, Nikhil Badugu, and Lu Wang. Argument mining for understanding peer reviews. In *Proc. of NAACL-HLT 2019*, page 2131–2137, 2019.

[236] Anthony Hunter. A probabilistic approach to modelling uncertain logical arguments. *Int. J. Approx. Reasoning*, 54(1):47–81, 2013.

[237] Anthony Hunter and Matthew Williams. Aggregating evidence about the positive and negative effects of treatments. *Artificial Intelligence in Medicine*, 56(3):173–190, 2012.

[238] Vijayasaradhi Indurthi, Bakhtiyar Syed, Manish Shrivastava, Nikhil Chakravartula, Manish Gupta, and Vasudeva Varma. FERMI at SemEval-2019 task 5: Using sentence embeddings to identify hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 70–74, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[239] Said Jai-Andaloussi, Imane El Mourabit, Nabil Madrane, Samia Benabdellah Chaouni, and Abderrahim Sekkaki. Soccer events summarization by using sentiment analysis. In *2015 International Conference on Computational Science and Computational Intelligence (CSCI)*, pages 398–403. IEEE, 2015.

[240] Mathilde Janier, John Lawrence, and Chris Reed. OVA+: an argument analysis interface. In *Computational Models of Argument - Proceedings of COMMA 2014, Atholl Palace Hotel, Scottish Highlands, UK, September 9-12, 2014*, pages 463–464, 2014.

[241] Soufian Jebbara, Valerio Basile, Elena Cabrio, and Philipp Cimiano. Extracting common sense knowledge via triple ranking using supervised and unsupervised distributional models. *Semantic Web*, 10(1):139–158, 2019.

[242] Rodolphe Jenatton, Nicolas L. Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 3167–3175. Curran Associates, Inc., 2012.

[243] Nanzhu Jiang and Meinard Müller. Estimating double thumbnails for music recordings. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 146–150, April 2015.

[244] Di Jin and Peter Szolovits. PICO element detection in medical text via long short-term memory neural networks. In *Proc. of BioNLP 2018 workshop*, pages 67–75, 2018.

[245] Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. Fasttext.zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*, 2016.

[246] Satya Katragadda, Ryan Benton, and Vijay Raghavan. Framework for real-time event detection using multiple social media sources. In *Proceedings of the 50th Hawaii International Conference on System Sciences*, 2017.

[247] Satya Katragadda, Shahid Virani, Ryan Benton, and Vijay Raghavan. Detection of event onset using twitter. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 1539–1546. IEEE, 2016.

[248] Jayong Kim and Y Yi Mun. A hybrid modeling approach for an automated lyrics-rating system for adolescents. In *European Conference on Information Retrieval*, pages 779–786. Springer, 2019.

[249] Florian Kleedorfer, Peter Knees, and Tim Pohle. Oh oh oh whoah! towards automatic topic detection in song lyrics. In *ISMIR*, 2008.

[250] Kevin Knight and Daniel Marcu. Statistics-based summarization - step one: Sentence compression. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 703–710. AAAI Press, 2000.

[251] Arne Köhn. What's in an embedding? Analyzing word embeddings through multilingual evaluation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 2067–2073, 2015.

[252] Alexandros Komninos and Suresh Manandhar. Dependency based embeddings for sentence classification tasks. In *Proc. of NAACL-HLT 2016*, pages 1490–1500, 2016.

[253] Parisa Kordjamshidi, Martijn Van Otterlo, and Marie-Francine Moens. Spatial role labeling: Towards extraction of spatial relations from natural language. *ACM Transactions on Speech and Language Processing (TSLP)*, 8(3):4, 2011. DOI: 10.1145/2050104.2050105.

[254] Efthymios Kouloumpis, Theresa Wilson, and Johanna D Moore. Twitter sentiment analysis: The good the bad and the omg! *Icwsm*, 11(538-541):164, 2011.

[255] M. Kouylekov and M. Negri. An open-source package for recognizing textual entailment. In *ACL System Demonstrations*, pages 42–47, 2010.

[256] Erwin Kreyszig. *Advanced engineering mathematics*. John Wiley & Sons, 2007.

[257] Rohan Kshirsagar, Tyrus Cukuvac, Kathy McKeown, and Susan McGregor. Predictive embeddings for hate speech detection on twitter. In *ALW2*, pages 26–32, 2018.

[258] Mitsumasa Kubo, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. Generating live sports updates from twitter by finding good reporters. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 01*, pages 527–534. IEEE Computer Society, 2013.

[259] Lars Kunze, Chris Burbridge, Marina Alberti, Akshaya Tippur, John Folkesson, Patric Jensfelt, and Nick Hawes. Combining top-down spatial reasoning and bottom-up object class recognition for scene understanding. In *IEEE IROS*, Chicago, Illinois, US, September, 14–18 2014.

[260] Gretel Liz De la Peña Sarracén, Reynaldo Gil Pons, Carlos Enrique Muñiz-Cuza, and Paolo Rosso. Hate speech detection using attention-based LSTM. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*, 2018.

[261] Loredana Laera, Ian Blacoe, Valentina Tamma, Terry Payne, Jérôme Euzenat, and Trevor Bench-Capon. Argumentation over ontology correspondences in MAS. In *Procs of AAMAS 2007*, pages 1–8, 2007.

[262] Thomas K. Landauer and Susan T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997. DOI: 10.1037/0033-295X.104.2.211.

[263] Thomas K. Landauer, Darrell Laham, Bob Rehder, and M. E. Schreiner. How well can passage meaning be derived without using word order? a comparison of latent semantic analysis and humans. In *Proc. of CSS*, pages 412–417, December 1997.

[264] J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.

[265] John Lawrence, Chris Reed, Colin Allen, Simon McAlister, and Andrew Ravenscroft. Mining arguments from 19th century philosophical texts using topic based modelling. In *Proceedings of ArgMin*, pages 79–87, 2014.

[266] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196, 2014.

[267] José Paulo Leal, Vânia Rodrigues, and Ricardo Queirós. Computing Semantic Relatedness using DBPedia. In Alberto Simões, Ricardo Queirós, and Daniela da Cruz, editors, *1st Symposium on Languages, Applications and Technologies*, volume 21 of *OpenAccess Series in Informatics (OASIcs)*, pages 133–147, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[268] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 2019.

[269] Younghun Lee, Seunghyun Yoon, and Kyomin Jung. Comparative studies of detecting abusive language on twitter. *CoRR*, abs/1808.10245, 2018.

[270] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. DBpedia - A large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 6(2):167–195, 2015. DOI: `10.3233/SW-140134`.

[271] Douglas B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):32–38, 1995. DOI: `10.1145/219717.219745`.

[272] Vladimir I Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966.

[273] Mark Levy, Mark Sandler, and Michael Casey. Extraction of high-level musical structure from audio data and its application to thumbnail generation. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 5, pages V–V. IEEE, 2006.

[274] Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni, and Noam Slonim. Context dependent claim detection. In *COLING*, pages 1489–1500, 2014.

[275] Jing Li, Aixin Sun, and Shafiq Joty. Segbot: A generic neural text segmentation model with pointer network. In *IJCAI*, pages 4166–4172, 2018.

[276] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, 2004.

[277] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2181–2187, 2015. DOI: `10.1016/j.procs.2017.05.045`.

[278] Marco Lippi and Paolo Torroni. Argument mining from speech: Detecting claims in political debates. In *AAAI*, pages 2979–2985, 2016.

[279] Marco Lippi and Paolo Torroni. Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Techn.*, 16(2):10, 2016.

[280] Marco Lippi and Paolo Torroni. Margot: A web server for argumentation mining. *Expert Systems with Applications*, 65:292–303, 12 2016.

[281] Bing Liu. *Sentiment Analysis and Opinion Mining*. Synthesis digital library of engineering and computer science. Morgan & Claypool, 2012.

[282] Hugo Liu and Push Singh. ConceptNet –A practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226, October 2004. DOI: 10.1023/B:BTTJ.0000047600.45421.6d.

[283] Ping Liu, Wen Li, and Liang Zou. NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 87–91, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[284] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[285] Clare Llewellyn, Claire Grover, Jon Oberlander, and Ewan Klein. Re-using an argument corpus to aid in the curation of social media collections. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014.*, pages 462–468, 2014.

[286] Markus Löchtefeld, Christian Jäckel, and Antonio Krüger. Twitsoccer: knowledge-based crowd-sourcing of live soccer events. In *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*, pages 148–151. ACM, 2015.

[287] Beth Logan, Andrew Kositsky, and Pedro Moreno. Semantic analysis of song lyrics. In *2004 IEEE International Conference on Multimedia and Expo (ICME) (IEEE Cat. No.04TH8763)*, volume 2, pages 827–830 Vol.2, June 2004.

[288] Luca Longo and Lucy Hederman. Argumentation theory for decision support in healthcare: A comparison with machine learning. In *Proc. of BHI 2013*, pages 168–180, 2013.

[289] Vanessa Lopez, Victoria S. Uren, Marta Sabou, and Enrico Motta. Cross ontology query answering on the semantic web: an initial evaluation. In *K-CAP*, pages 17–24, 2009.

[290] Vanessa Lopez, Victoria S. Uren, Marta Sabou, and Enrico Motta. Is question answering fit for the semantic web?: A survey. *Semantic Web*, 2(2):125–155, 2011.

[291] Annie Louis and Ani Nenkova. Automatically assessing machine summary content without a gold standard. *Computational Linguistics*, 39(2), 2013.

[292] Max Lungarella, Giorgio Metta, Rolf Pfeifer, and Giulio Sandini. Developmental robotics: a survey. *Connection Science*, 15(4):151–190, 2003. DOI: 10.1080/09540090310001655110.

[293] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*, 2016.

[294] Stuart Mackie, Richard McCreadie, Craig Macdonald, and Iadh Ounis. On choosing an effective automatic evaluation metric for microblog summarisation. In *Proceedings of the 5th Information Interaction in Context Symposium*, pages 115–124. ACM, 2014.

[295] Bernardo Magnini, Roberto Zanoli, Ido Dagan, Kathrin Eichler, Günter Neumann, Tae-Gil Noh, Sebastian Pado, Asher Stern, and Omer Levy. The excitement open platform for textual inferences. In *Proceedings of ACL (System Demonstrations)*, pages 43–48, 2014.

[296] Jose P. G. Mahedero, Álvaro Martínez, Pedro Cano, Markus Koppenberger, and Fabien Gouyon. Natural language processing of lyrics. In *Proceedings of the 13th Annual ACM International Conference on Multimedia*, MULTIMEDIA '05, pages 475–478, New York, NY, USA, 2005. ACM.

[297] Rahmad Mahendra, Lilian Wanzare, Bernardo Magnini, Raffaella Bernardi, and Alberto Lavelli. Acquiring relational patterns from wikipedia: A case study. In *LTC2011*, 11 2011.

[298] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL (System Demonstrations)*, pages 55–60, 2014.

[299] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 227–236. ACM, 2011.

[300] Marie-Catherine De Marneffe, Anne N. Rafferty, and Christopher D. Manning. Finding contradictions in text. In *Proc. of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, June 2008.

[301] Rudolf Mayer and Andreas Rauber. Musical genre classification by ensembles of audio and lyrics features. In *Proceedings of the 12th International Conference on Music Information Retrieval*, pages 675–680, 2011.

[302] Tobias Mayer, Elena Cabrio, Marco Lippi, Paolo Torroni, and Serena Villata. Argument mining on clinical trials. In Sanjay Modgil, Katarzyna Budzynska, and John Lawrence, editors, *Computational Models of Argument - Proceedings of COMMA 2018, Warsaw, Poland, 12-14 September 2018*, volume 305 of *Frontiers in Artificial Intelligence and Applications*, pages 137–148. IOS Press, 2018.

[303] Tobias Mayer, Elena Cabrio, and Serena Villata. ACTA A tool for argumentative clinical trial analysis. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 6551–6553, 2019.

[304] Tobias Mayer, Elena Cabrio, and Serena Villata. Transformer-based argument mining for healthcare applications. In *To appear in Proceedings of the 24th European Conference on Artificial Intelligence (ECAI2020)*, 2020.

[305] Julian J. McAuley, Rahul Pandey, and Jure Leskovec. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10-13, 2015*, pages 785–794, 2015. DOI: 10.1145/2783258.2783381.

[306] John McCarthy. Circumscription - A form of non-monotonic reasoning. *Artificial Intelligence*, 13(1-2):27–39, 1980. DOI: 10.1016/0004-3702(80)90011-9.

[307] Andrew J McMinn and Joemon M Jose. Real-time entity-based event detection for twitter. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 65–77. Springer, 2015.

[308] Andrew J. McMinn, Yashar Moshfeghi, and Joemon M Jose. Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of CIKM*, pages 409–418. ACM, 2013.

[309] Qiaozhu Mei and ChengXiang Zhai. Generating impact-based summaries for scientific literature. In *ACL*, 2008.

[310] Pablo Mendes, Max Jakob, and Christian Bizer. DBpedia: A multilingual cross-domain knowledge base. In *Procs of LREC 2012*. ELRA, 2012.

[311] Pablo Mendes, Hannes Mühleisen, and Christian Bizer. Sieve: linked data quality assessment and fusion. In *Procs of the Joint EDBT/ICDT Workshops*, pages 116–123. ACM, 2012.

[312] Stefano Menini, Elena Cabrio, Sara Tonelli, and Serena Villata. Never retreat, never retract: Argumentation analysis for political speeches. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18)*, pages 4889–4896, 2018.

[313] Stefano Menini, Giovanni Moretti, Michele Corazza, Elena Cabrio, Sara Tonelli, and Serena Villata. A system to monitor cyberbullying based on message classification and social network analysis. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 105–110, Florence, Italy, August 2019. Association for Computational Linguistics.

[314] Stefano Menini, Federico Nanni, Simone Paolo Ponzetto, and Sara Tonelli. Topic-based agreement and disagreement in us electoral manifestos. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2928–2934, Stroudsburg, PA, September 2017. Association for Computational Linguistics.

[315] Stefano Menini and Sara Tonelli. Agreement and disagreement: Comparison of points of view in the political domain. In *Proceedings of COLING*, 2016.

[316] Gabriel Meseguer-Brocal, Alice Cohen-Hadria, and Geoffroy Peeters. Dali: a large dataset of synchronized audio, lyrics and notes, automatically created using teacher-student machine learning paradigm. In *ISMIR Paris, France*, 2018.

[317] Gabriel Meseguer-Brocal, Geoffroy Peeters, Guillaume Pellerin, Michel Buffa, Elena Cabrio, Catherine Faron Zucker, Alain Giboin, Isabelle Mirbel, Romain Hennequin, Manuel Moussallam, Francesco Piccoli, and Thomas Fillon. WASABI: a Two Million Song Database Project with Audio and Cultural Metadata plus WebAudio enhanced Client Applications. In *Web Audio Conference 2017 – Collaborative Audio #WAC2017*, London, United Kingdom, August 2017. Queen Mary University of London.

[318] Rada Mihalcea and Carlo Strapparava. Lyrics, music, and emotions. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 590–599, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[319] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[320] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[321] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[322] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, 2013.

[323] Guido Minnen, John A. Carroll, and Darren Pearce. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223, 2001. DOI: 10.1017/S1351324901002728.

[324] Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. Neural character-based composition models for abuse detection. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 1–10, Brussels, Belgium, October 2018. Association for Computational Linguistics.

[325] Dipendra K. Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 35(1-3):281–300, 2016. DOI: 10.1177/0278364915602060.

[326] Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 236–244, 2008. DOI: 10.1039/9781847558633-00236.

[327] Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha Pratim Talukdar, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matthew Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapandula Nakashole, Emmanouil Antonios Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Tanti Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. Never-ending learning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2302–2310, 2015.

[328] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. In *Proc. of ACL 2016*, pages 1105–1116, 2016.

[329] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*, 2012.

[330] Raquel Mochales and Marie-Francine Moens. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22, 2011.

[331] Saif Mohammad. Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 174–184, 2018.

[332] Saif Mohammad, Felipe Bravo-Marquez, Mohammad Salameh, and Svetlana Kiritchenko. Semeval-2018 task 1: Affect in tweets. In *Proceedings of the 12th international workshop on semantic evaluation*, pages 1–17, 2018.

[333] Saif Mohammad and Peter D Turney. Emotions evoked by common words and phrases: Using mechanical turk to create an emotion lexicon. In *Proceedings of the NAACL HLT 2010 workshop on computational approaches to analysis and generation of emotion in text*, pages 26–34. Association for Computational Linguistics, 2010.

[334] Saif Mohammad and Peter D Turney. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, 29(3):436–465, 2013.

[335] Christof Monz and Maarten de Rijke. Light-weight entailment checking for computational semantics. In *Proc. Inference in Computational Semantics (ICoS-3)*, pages 59–72, June 2001.

[336] Raymond J. Mooney. Learning to connect language and perception. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pages 1598–1601, 2008.

[337] Giovanni Moretti, Rachele Sprugnoli, and Sara Tonelli. Digging in the Dirt: Extracting Keyphrases from Texts with KD. In *Proceedings of the Second Italian Conference on Computational Linguistics*, 2015.

[338] Nona Naderi and Graeme Hirst. Argumentation mining in parliamentary discourse. In *CMNA*, pages 16–25, 2015.

[339] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, 2016.

[340] Barak Naveh et al. Jgrapht. *Internet: http://jgrapht.sourceforge.net*, 2008.

[341] Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217 – 250, 2012.

[342] Ani Nenkova, Kathleen McKeown, et al. Automatic summarization. *Foundations and Trends® in Information Retrieval*, 5(2–3):103–233, 2011.

[343] Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, pages 189–198. ACM, 2012.

[344] Vlad Niculae, Joonsuk Park, and Claire Cardie. Argument mining with structured SVMs and RNNs. In *Proc. of ACL 2017*, pages 985–995, 2017.

[345] Chikashi Nobata, Joel R. Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *WWW*, pages 145–153, 2016.

[346] Farhad Nooralahzadeh, Cédric Lopez, Elena Cabrio, Fabien L. Gandon, and Frédérique Segond. Adapting semantic spreading activation to entity linking in text. In *Natural Language Processing and Information Systems - 21st International Conference on Applications of Natural Language to Information Systems, NLDB 2016, Salford, UK, June 22-24, 2016, Proceedings*, pages 74–90, 2016.

[347] Farid Nouioua and Vincent Risch. Bipolar argumentation frameworks with specialized supports. In *Proc. of the 22nd IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 215–218. IEEE Computer Society, October 2010.

[348] Farid Nouioua and Vincent Risch. Argumentation frameworks with necessities. In *Proc. of the 5th International Conference Scalable Uncertainty Management (SUM), LNCS 6929*, pages 163–176, October 2011.

[349] Robertus Nugroho, Weiliang Zhao, Jian Yang, Cecile Paris, Surya Nepal, and Yan Mei. Time-sensitive topic derivation in twitter. In *International Conference on Web Information Systems Engineering*, pages 138–152. Springer, 2015.

[350] Brendan O'Connor, Michel Krieger, and David Ahn. Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM*, pages 384–385, 2010.

[351] Markus Ojala and Gemma C. Garriga. Permutation tests for studying classifier performance. *J. Mach. Learn. Res.*, 11:1833–1863, August 2010.

[352] Nathan Ong, Diane Litman, and Alexandra Brusilovsky. Ontology-based argument mining and automatic essay scoring. In *Proceedings of the First Workshop on Argumentation Mining*, pages 24–28, 2014.

[353] Nir Oren and Timothy J. Norman. Semantics for evidence-based argumentation. In *Proc. of the International Conference on Computational Models of Argument (COMMA), Frontiers in Artificial Intelligence and Applications 172*, pages 276–284, May 2008.

[354] Nir Oren, Chris Reed, and Michael Luck. Moving between argumentation frameworks. In *Proc. of the International Conference on Computational Models of Argument (COMMA), Frontiers in Artificial Intelligence and Applications 216*, pages 379–390, September 2010.

[355] Jahna Otterbacher, Güneş Erkan, and Dragomir R Radev. Using random walks for question-focused sentence retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 915–922. Association for Computational Linguistics, 2005.

[356] Selami Özsoy. Use of new media by turkish fans in sport communication: Facebook and twitter. *Journal of Human Kinetics*, 28:165–176, 2011.

[357] Joaquin Padilla Montani and Peter Schüller. Tuwienkbs at germeval 2018: German abusive tweet detection. In *GermEval*, 09 2018.

[358] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[359] Sandeep Panem, Manish Gupta, and Vasudeva Varma. Structured information extraction from natural disaster events on twitter. In *Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, pages 1–8. ACM, 2014.

[360] Patrick Pantel and Marco Pennacchiotti. Espresso: Leveraging generic patterns for automatically harvesting semantic relations. In Nicoletta Calzolari, Claire Cardie, and Pierre Isabelle, editors, *ACL 2006, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 17-21 July 2006*. The Association for Computer Linguistics, 2006. DOI: 10.3115/1220175.1220190.

[361] Ruchi Parikh and Kamalakar Karlapalem. Et: events from tweets. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 613–620. ACM, 2013.

[362] Loreto Parisi, Simone Francia, Silvio Olivastri, and Maria Stella Tavella. Exploiting synchronized lyrics and vocal features for music emotion detection. *CoRR*, abs/1901.04831, 2019.

[363] Joonsuk Park, Cheryl Blake, and Claire Cardie. Toward machine-assisted participation in erulemaking: an argumentation model of evaluability. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law, ICAIL 2015, San Diego, CA, USA, June 8-12, 2015*, pages 206–210, 2015.

[364] Joonsuk Park and Claire Cardie. Identifying appropriate support for propositions in online user comments. In *Proceedings of the ArgMin*, pages 29–38, 2014.

[365] Daraksha Parveen, Hans-Martin Ramsl, and Michael Strube. Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1949–1954, 2015.

[366] Daraksha Parveen and Michael Strube. Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 1298–1304. AAAI Press, 2015.

[367] J. Pauwels and M. Sandler. A web-based system for suggesting new practice material to music learners based on chord content. In *Joint Proc. 24th ACM IUI Workshops (IUI2019)*, 2019.

[368] J. Pauwels, A. Xambó, G. Roma, M. Barthet, and G. Fazekas. Exploring real-time visualisations to support chord learning with a large music collection. In *Proc. 4th Web Audio Conf. (WAC 2018)*, 2018.

[369] Samuel Pecar. Towards opinion summarization of customer reviews. In *Proceedings of ACL 2018, Student Research Workshop*, pages 1–8. Association for Computational Linguistics, 2018.

[370] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[371] Andreas Peldszus and Manfred Stede. From argument diagrams to argumentation mining in texts: A survey. *Int. J. Cogn. Inform. Nat. Intell.*, 7(1):1–31, 2013.

[372] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proc. of EMNLP 2014*, pages 1532–1543, 2014.

[373] Juan Manuel Pérez and Franco M. Luque. Atalaya at SemEval 2019 task 5: Robust embeddings for tweet classification. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 64–69, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[374] Isaac Persing and Vincent Ng. End-to-end argumentation mining in student essays. In *Proc. of NAACL-HLT 2016*, pages 1384–1394, 2016.

[375] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL-HLT 2018*, pages 2227–2237, 2018.

[376] Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 181–189. Association for Computational Linguistics, 2010.

[377] Saša Petrović, Miles Osborne, and Victor Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 338–346, 2012.

[378] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR 2008*, pages 1–8, June 2008.

[379] Lawrence Philips. The double metaphone search algorithm. *C/C++ Users Journal*, 18:38–43, 06 2000.

[380] Emanuele Pianta, Luisa Bentivogli, and Christian Girardi. Multiwordnet: developing an aligned multilingual database. In *Proceedings of the First International Conference on Global WordNet*, January 2002.

[381] Manfred Pinkal. Logic and lexicon: the semantics of the indefinite. *Studies in linguistics and philosophy*, 56, 1995.

[382] Marco Polignano and Pierpaolo Basile. Hansel: Italian hate speech detection through ensemble learning and deep neural networks. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*, 2018.

[383] Ana-Maria Popescu, Marco Pennacchiotti, and Deepa Paranjpe. Extracting events and event descriptions from twitter. In *Proceedings of the 20th international conference companion on World wide web*, pages 105–106. ACM, 2011.

[384] Peter Potash, Alexey Romanov, and Anna Rumshisky. Here's my point: Joint pointer architecture for argument mining. In *Proc. of EMNLP 2017*, pages 1364–1373, 2017.

[385] Henry Prakken. An abstract framework for argumentation with structured arguments. *Argument & Computation*, 1:93–124, 2010.

[386] Michal Ptaszynski, Fumito Masui, Yasutomo Kimura, Rafal Rzepka, and Kenji Araki. Automatic extraction of harmful sentence patterns with application in cyberbullying detection. In *Human Language Technology. Challenges for Computer Science and Linguistics - 7th Language and Technology Conference, LTC 2015, Poznań, Poland, November 27-29, 2015, Revised Selected Papers*, pages 349–362, 2015.

[387] Malik Al Qassas, Daniela Fogli, Massimiliano Giacomin, and Giovanni Guida. Analysis of clinical discussions based on argumentation schemes. *Procedia Computer Science*, 64:282–289, 2015.

[388] Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. A word at a time: computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web, WWW 2011, Hyderabad, India, March 28 - April 1, 2011*, pages 337–346, 2011. DOI: 10.1145/1963405.1963455.

[389] Iyad Rahwan and Guillermo Simari, editors. *Argumentation in Artificial Intelligence*. Springer, 2009.

[390] C. Reed and F. Grasso. Recent advances in computational models of natural argument. *Int. J. Intell. Syst.*, 22(1):1–15, 2007.

[391] Chris Reed and Glen Rowe. Araucaria: Software for argument analysis, diagramming and representation. *Int. Journal on Artificial Intelligence Tools*, 13(4):961–980, 2004.

[392] Michaela Regneri, Alexander Koller, and Manfred Pinkal. Learning script knowledge with web experiments. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 979–988, 2010.

[393] Nils Reimers, Benjamin Schiller, Tilman Beck, Johannes Daxenberger, Christian Stab, and Iryna Gurevych. Classification and clustering of arguments with contextualized word embeddings. In *Proc. of ACL 2019*, pages 567–578, 2019.

[394] Joseph Reisinger and Raymond J. Mooney. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 109–117, 2010.

[395] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. Relation extraction with matrix factorization and universal schemas. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 74–84. The Association for Computational Linguistics, 2013.

[396] Ruty Rinott, Lena Dankin, Carlos Alzate, Mitesh M. Khapra, Ehud Aharoni, and Noam Slonim. Show me your evidence - an automatic method for context dependent evidence detection. In *EMNLP*, pages 440–450, 2015.

[397] Ana Rojo. *Step by Step: A Course in Contrastive Linguistics and Translation*. Peter Lang, 2009.

[398] Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. Investigating a generic paraphrase-based approach for relation extraction. In *Proc. of the 11st Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 409–416, April 2006.

[399] James A Russell. A circumplex model of affect. *Journal of personality and social psychology*, 39(6):1161, 1980.

[400] Horacio Saggion and Thierry Poibeau. *Automatic Text Summarization: Past, Present and Future*, pages 3–21. Springer, Berlin, Heidelberg, 01 2013.

[401] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, pages 851–860. ACM, 2010.

[402] Muhammad Saleem, Samaneh Nazari Dastjerdi, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. Question answering over linked data: What is difficult to answer? what affects the f scores? In *Natural Language Interfaces workshop at ISWC*, 2017.

[403] Valentino Santucci, Stefania Spina, Alfredo Milani, Giulio Biondi, and Gabriele Di Bari. Detecting hate speech for italian language in social media. In *Proceedings of the Sixth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2018) co-located with the Fifth Italian Conference on Computational Linguistics (CLiC-it 2018), Turin, Italy.*, 2018.

[404] Enrico Santus, Emmanuele Chersoni, Alessandro Lenci, Chu-Ren Huang, and Philippe Blache. Testing apsyn against vector cosine on similarity estimation. In *Proceedings of the 30th Pacific Asia Conference on Language, Information and Computation, PACLIC 30, Seoul, Korea, October 28 - October 30, 2016*, pages 229–238, 2016.

[405] Ashutosh Saxena, Ashesh Jain, Ozan Sener, Aditya Jami, Dipendra Kumar Misra, and Hema Swetha Koppula. Robobrain: Large-scale knowledge engine for robots. *CoRR*, abs/1412.0691, 2014.

[406] Tatjana Scheffler, Erik Haegert, Santichai Pornavalaia, and Mino Lee Sasse. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[407] Martin J. Schuster, Dominik Jain, Moritz Tenorth, and Michael Beetz. Learning organizational principles in human environments. In *ICRA*, pages 3867–3874, May 2012.

[408] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 1997.

[409] Elaheh ShafieiBavani, Mohammad Ebrahimi, Raymond Wong, and Fang Chen. Summarization evaluation in the absence of human model summaries using the compositionality of word embeddings. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 905–914. Association for Computational Linguistics, 2018.

[410] Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 685–688. Association for Computational Linguistics, 2010.

[411] Abhishek Singh, Eduardo Blanco, and Wei Jin. Incorporating emoji descriptions improves tweet classification. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[412] Peter K. Smith, Jess Mahdavi, Manuel Carvalho, Sonja Fisher, Shanette Russell, and Neil Tippett. Cyberbullying: its nature and impact in secondary school pupils. *Journal of Child Psychology and Psychiatry*, 49(4):376–385, 2008.

[413] Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. In *ICLR*, 2017.

[414] Jan Snajder. Social media argumentation mining: The quest for deliberateness in raucousness. *CoRR*, abs/1701.00168, 2017.

[415] Cees GM Snoek and Marcel Worring. Multimedia event-based video indexing using time intervals. *IEEE Transactions on Multimedia*, 7(4):638–647, 2005.

[416] Cees GM Snoek, Marcel Worring, et al. Time interval based modelling and classification of events in soccer video. In *Proceedings of the 9th annual conference of the advanced school for computing and imaging (ASCI), Heijen*. Citeseer, 2003.

[417] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning with neural tensor networks for knowledge base completion. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 926–934, 2013.

[418] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, volume 1631, page 1642, 2013.

[419] Shuran Song, Linguang Zhang, and Jianxiong Xiao. Robot in a room: Toward perfect object recognition in closed environments. *CoRR*, abs/1507.02703, 2015.

[420] Rachele Sprugnoli, Stefano Menini, Sara Tonelli, Filippo Oncini, and Enrico Piras. Creating a whatsapp dataset to study pre-teen cyberbullying. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 51–59. Association for Computational Linguistics, 2018.

[421] Rachele Sprugnoli and Sara Tonelli. One, no one and one hundred thousand events: Defining and processing events in an inter-disciplinary perspective. *Natural Language Engineering*, 23(4):485–506, 2017.

[422] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[423] Christian Stab, Johannes Daxenberger, Chris Stahlhut, Tristan Miller, Benjamin Schiller, Christopher Tauchmann, Steffen Eger, and Iryna Gurevych. ArgumenText: Searching for arguments in heterogeneous sources. In *Proceedings of NAACL (demo)*, June 2018.

[424] Christian Stab and Iryna Gurevych. Annotating argument components and relations in persuasive essays. In *COLING*, pages 1501–1510, 2014.

[425] Christian Stab and Iryna Gurevych. Parsing argumentation structures in persuasive essays. *Computational Linguistics*, 43(3):619–659, 2017.

[426] Jacopo Staiano and Marco Guerini. Depeche mood: a lexicon for emotion analysis from crowd annotated news. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 427–433, 2014.

[427] Dominik Stammbach, Azin Zahraei, Polina Stadnikova, and Dietrich Klakow. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[428] Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. Brat: a web-based tool for nlp-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 102–107. Association for Computational Linguistics, 2012.

[429] Lucas Sterckx. *Topic detection in a million songs*. PhD thesis, PhD thesis, Ghent University, 2014.

[430] Fabian-Robert Stöter, Stefan Uhlich, Antoine Liutkus, and Yuki Mitsufuji. Open-unmix-a reference implementation for music source separation. *Journal of Open Source Software*, 2019.

[431] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Procs of WWW 2007*, pages 697–706. ACM, 2007.

[432] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. Multi-instance multi-label learning for relation extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 455–465. ACL, 2012.

[433] Philip Tagg. Analysing popular music: theory, method and practice. *Popular Music*, 2:37–67, 1982.

[434] Leonard Talmy. The fundamental system of spatial schemas in language. *From perception to meaning: Image schemas in cognitive linguistics*, 3, 2005. DOI: 10.1515/9783110197532.3.199.

[435] Moritz Tenorth and Michael Beetz. Knowrob: A knowledge processing infrastructure for cognition-enabled robots. *The International Journal of Robotics Research*, 32(5):566–590, 2013. DOI: 10.1177/0278364913481635.

[436] Moritz Tenorth, Lars Kunze, Dominik Jain, and Michael Beetz. KNOWROB-MAP – knowledge-linked semantic object maps. In *IEEE-RAS ICHR*, pages 430–435, Nashville, TN, USA, December 6-8 2010.

[437] Moritz Tenorth, Daniel Nyga, and Michael Beetz. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *IEEE International Conference on Robotics and Automation, ICRA 2010, Anchorage, Alaska, USA, 3-7 May 2010*, pages 1486–1491, 2010. DOI: 10.1109/ROBOT.2010.5509955.

[438] Simone Teufel, Advaith Siddharthan, and Colin Batchelor. Towards domain-independent argumentative zoning: Evidence from chemistry and computational linguistics. In *Proc. of EMNLP 2009*, pages 1493–1502, 2009.

[439] Akshaya Thippur, Chris Burbridge, Lars Kunze, Marina Alberti, John Folkesson, Patric Jensfelt, and Nick Hawes. A comparison of qualitative and metric spatial relation models for scene understanding. In *AAAI'15*, January 2015.

[440] Robert S. Tokunaga. Following you home from school: A critical review and synthesis of research on cyberbullying victimization. *Computers in Human Behavior*, 26(3):277 – 287, 2010.

[441] Sabina Tomkins, Lise Getoor, Yunfei Chen, and Yi Zhang. A socio-linguistic model for cyberbullying detection. In *IEEE/ACM 2018 International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018, Barcelona, Spain, August 28-31, 2018*, pages 53–60, 2018.

[442] Stephen Toulmin. *The Uses of Argument*. Cambridge University Press, 1958.

[443] Antonio Trenta, Anthony Hunter, and Sebastian Riedel. Extraction of evidence tables from abstracts of randomized clinical trials using a maximum entropy classifier and global constraints. *CoRR*, abs/1509.05209, 2015.

[444] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proc. of ICML 2016*, pages 2071–2080, 2016.

[445] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based qa over rdf data. In *Procs of WWW 2012*, pages 639–648, 2012.

[446] Christina Unger and Philipp Cimiano. Pythia: Compositional meaning construction for ontology-based question answering on the semantic web. In *Procs of NLDB 2011*, pages 153–160, 2011.

[447] Christina Unger, Corina Forascu, Vanessa López, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. Question answering over linked data (QALD-4). In *Working Notes for CLEF 2014 Conference, Sheffield, UK, September 15-18, 2014*, pages 1172–1180, 2014.

[448] Christina Unger, Corina Forascu, Vanessa López, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. Question answering over linked data (QALD-5). In *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015*, 2015.

[449] Christina Unger, André Freitas, and Philipp Cimiano. *An Introduction to Question Answering over Linked Data*, pages 100–140. Springer International Publishing, Cham, 2014.

[450] Christina Unger, Axel-Cyrille Ngonga Ngomo, and Elena Cabrio. 6th open challenge on question answering over linked data (QALD-6). In *Semantic Web Challenges - Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 171–177, 2016.

[451] Ricardo Usbeck, Ria Gusmita, Muhammad Saleem, and Axel-Cyrille Ngonga Ngomo. 9th challenge on question answering over linked data (qald-9). 11 2018.

[452] Marieke Van Erp, Giuseppe Rizzo, and Raphaël Troncy. Learning with the web: Spotting named entities on the intersection of nerd and machine learning. In *# MSM*, pages 27–30. Citeseer, 2013.

[453] Laurent Vanni, Mélanie Ducoffe, Carlos Aguilar, Frederic Precioso, and Damon Mayaffre. Textual deconvolution saliency (tds): a deep tool box for linguistic analysis. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 548–557, 2018.

[454] Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80. Association for Computational Linguistics, 2007.

[455] Bart Verheij. Argumed - a template-based argument mediation system for lawyers and legal knowledge based systems. In *Proc. of the 11th International Conference on Legal Knowledge and Information Systems (JURIX)*, pages 113–130, December 1998.

[456] Maria Paz Garcia Villalba and Patrick Saint-Dizier. Some facets of argument mining for opinion analysis. In *Proceedings of the 2012 Conference on Computational Models of Argument*, pages 23–34, 2012.

[457] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[458] Serena Villata, Elena Cabrio, Imène Jraidi, Sahbi Benlamine, Maher Chaouachi, Claude Frasson, and Fabien Gandon. Emotions and personality traits in argumentation: An empirical evaluation. *Argument & Computation*, 8(1):61–87, 2017.

[459] Jacky Visser, Barbara Konat, Rory Duthie, Marcin Koszowy, Katarzyna Budzynska, and Chris Reed. Argumentation in the 2016 us presidential elections: annotated corpora of television debates and social media reaction. *Language Resources and Evaluation*, 2019.

[460] Dirk von Grunigen, Ralf Grubenmann, Fernando Benites, Pius Von Daniken, and Mark Cieliebak. spmmmp at germeval 2018 shared task: Classification of offensive content in tweets using convolutional neural networks and gated recurrent units. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[461] Henning Wachsmuth, Shahbaz Syed, and Benno Stein. Retrieval of the best counterargument without prior topic knowledge. In *Proceedings of ACL*, pages 241–251, 2018.

[462] Markus Waibel, Michael Beetz, Raffaello D'Andrea, Rob Janssen, Moritz Tenorth, Javier Civera, Jos Elfring, Dorian Gálvez-López, Kai Häussermann, J.M.M. Montiel, Alexander Perzylo, Björn Schießle, Oliver Zweigle, and René van de Molengraft. RoboEarth - A World Wide Web for Robots. *Robotics & Automation Magazine*, 18(2):69–82, 2011. DOI: 10.1109/MRA.2011.941632.

[463] Marylin Walker, Jean Fox Tree, Pranav Anand, R. Abbott, and Joseph King. A corpus for research on deliberation and debate. In *LREC*, pages 812–817, 2012.

[464] Douglas Walton. Explanations and arguments based on practical reasoning. In Thomas Roth-Berghofer, Nava Tintarev, and David B. Leake, editors, *Explanation-aware Computing, Papers from the 2009 IJCAI Workshop, Pasadena, California , USA, July 11-12, 2009*, pages 72–83, 2009.

[465] Douglas Walton, Christopher Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

[466] Lu Wang and Claire Cardie. Improving agreement and disagreement identification in online discussions with A socially-tuned sentiment lexicon. In *Proceedings of the 5th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, WASSA@ACL 2014, June 27, 2014, Baltimore, Maryland, USA*, pages 97–106, 2014.

[467] Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, and Claire Cardie. A sentence compression based framework to query-focused multi-document summarization. *arXiv preprint arXiv:1606.07548*, 2016.

[468] Xiaofeng Wang, Matthew S Gerber, and Donald E Brown. Automatic crime prediction using events extracted from twitter posts. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, pages 231–238. Springer, 2012.

[469] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 1112–1119, 2014.

[470] Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior research methods*, 45(4):1191–1207, 2013.

[471] Zeerak Waseem, Wendy Hui Kyong Chung, Dirk Hovy, and Joel Tetreault. Proceedings of the first workshop on abusive language online. In *Proceedings of the First Workshop on Abusive Language Online*. Association for Computational Linguistics, 2017.

[472] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *SRW@HLT-NAACL*, 2016.

[473] Kento Watanabe, Yuichiroh Matsubayashi, Naho Orita, Naoaki Okazaki, Kentaro Inui, Satoru Fukayama, Tomoyasu Nakano, Jordan Smith, and Masataka Goto. Modeling discourse segments in lyrics using repeated patterns. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1959–1969, 2016.

[474] Gregor Wiedeman, Eugen Ruppert, Raghav Jindal, and Chris Biemann. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[475] Michael Wiegand, Anastasija Amann, Tatiana Anikina, Aikaterini Azoidou, Anastasia Borisenkov, Kirstin Kolmorgen, Insa Kroger, and Christine Schafer. Overview of the germeval 2018 shared task on the identification of offensive language. In *Proceedings of GermEval 2018, 14th Conference on Natural Language Processing (KONVENS 2018)*, 2018.

[476] Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[477] Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. Overview of the germeval 2018 shared task on the identification of offensive language. In *GermEval 2018*, 2018.

[478] Fei Wu and Daniel S. Weld. Open information extraction using Wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127, Uppsala, Sweden, July 2010. Association for Computational Linguistics.

[479] Zhibiao Wu and Martha Palmer. Verbs semantics and lexical selection. In *ACL*, ACL '94, pages 133–138, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics.

[480] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. In *WWW*, pages 1391–1399, 2017.

[481] Adam Wyner, Jodi Schneider, Katie Atkinson, and Trevor J. Bench-Capon. Semi-automated argumentative analysis of online product reviews. In *Proceedings of the 2012 Conference on Computational Models of Argument*, pages 43–50, 2012.

[482] Yunqing Xia, Linlin Wang, Kam-Fai Wong, and Mingxing Xu. Sentiment vector space model for lyric-based song sentiment classification. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 133–136, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[483] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 3485–3492, 2010. DOI: 10.1109/CVPR.2010.5539970.

[484] Wei Xu, Ralph Grishman, Adam Meyers, and Alan Ritter. A preliminary study of tweet summarization using information extraction. *NAACL 2013*, page 20, 2013.

[485] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *ICLR*, 2015.

[486] Dan Yang and Won-Sook Lee. Music emotion identification from lyrics. In *2009 11th IEEE International Symposium on Multimedia*, pages 624–629, Dec 2009.

[487] Yiming Yang, Thomas Pierce, and Jaime G. Carbonell. A study of retrospective and on-line event detection. In *SIGIR '98*, 1998.

[488] Dawei Yin, Zhenzhen Xue, Liangjie Hong, Brian D. Davison, April Kontostathis, and Lynne Edwards. Detection of harassment on web 2.0. In *Proceedings of the Content Analysis in the Web*, pages 1–7, 2009.

[489] Jay Young, Valerio Basile, Lars Kunze, Elena Cabrio, and Nick Hawes. Towards lifelong object learning by integrating situated robot perception and semantic web mining. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, pages 1458–1466, 2016.

[490] Jay Young and Nick Hawes. Learning by observation using qualitative spatial relations. In *AAMAS 2015*, May 2015.

[491] Jay Young, Lars Kunze, Valerio Basile, Elena Cabrio, Nick Hawes, and Barbara Caputo. Semantic web-mining and deep vision for lifelong object discovery. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pages 2774–2779, 2017.

[492] Jure Zabkar, Martin Mozina, Jerneja Videcnik, and Ivan Bratko. Argument based machine learning in a medical domain. In *Proc. of COMMA 2006*, pages 59–70, 2006.

[493] Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[494] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *Proc. of EMNLP 2018*, pages 93–104, 2018.

[495] Robinson-D. Zhang, Z. and J. Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *ESWC*, pages 745–760. Springer Verlag, 2018.

[496] Deyu Zhou, Liangyu Chen, and Yulan He. A simple bayesian modelling approach to event extraction from twitter. *Atlantis*, page 0, 2011.

[497] Deyu Zhou, Xuan Zhang, and Yulan He. Event extraction from Twitter using Non-Parametric Bayesian Mixture Model with Word Embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 808–817, Valencia, Spain, April 2017.

[498] Valentino Zurloni and Luigi Anolli. Fallacies as argumentative devices in political debates. In *Multimodal Communication in Political Speech. Shaping Minds and Social Action - International Workshop, Political Speech 2010, Rome, Italy, November 10-12, 2010, Revised Selected Papers*, pages 245–257, 2010.