Personalized, Aspect-based Summarization of Movie Reviews

Sara El Aouad

UNIVERSITY OF PIERRE AND MARIE CURIE

DOCTORAL THESIS

Personalized, Aspect-based Summarization of Movie Reviews

Author:
Sara EL AOUAD

Advisors:
Renata Teixeira, Vassilis Christophides, Christophe Diot

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

MIMOVE team
INRIA

April 16, 2019

Declaration of Authorship

I, Sara EL AOUAD, declare that this thesis titled, "Personalized, Aspect-based Summarization of Movie Reviews " and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Abstract

Online reviewing has become a key resource to help users decide what to buy and watch or places to go. A number of platforms allow users to express their opinions about various products or services using numerical ratings as well as textual comments (for example, Trip Advisor[1] for hotels or IMDB[2] for movies). The numerical ratings give a coarse idea of whether the service or product is good or bad, without further explanation behind the rating. On the other hand, textual comments give full details, but reading all comments is time-consuming and tedious for users. This dissertation argues for summarizing textual comments in a concise text that captures the main opinions in the set of reviews. We focus on online reviews of movies that the literature considers more challenging than mining other types of reviews, such as product reviews.

Unfortunately, a single summary is unlikely to serve all users in a large population as prior studies have showed that users tend to have different interests regarding the *aspects* of products or services. Aspects, also called features, describe the characteristics of a product that users give opinions about, i.e., "Story" and "Visual effects" for movies. For example, a user might be interested in the actor performance, while another user in the story of the movie. Most previous studies on aspect-based summary generation have focused on extracting information that summarizes the opinions about aspects in general, without considering the personal preferences of the user.

In this dissertation, we develop novel methods and algorithms to generate personalized, aspect-based summaries of the reviews of a given movie for a given user. We rely on a large dataset with movie reviews from the IMDB website to address three important problems of personalized aspect-based summarization.

The first problem is to extract the "signature" of an aspect, or the set of words that are related to the aspect, from movie reviews. We develop an algorithm that automatically extracts the signature of each aspect. Our approach simplifies this process as it works in one step: we extract and group related terms to a seed term of each aspect at the same time. Our evaluation shows that our method is able to extract even unpopular terms that represent an aspect, such as compound terms or abbreviations, as opposed to the methods proposed in the related work, which focus on popular terms alone.

We then study the problem of annotating sentences with aspects, and propose a new method that annotates sentences based on a similarity between the aspect signature and the terms in the sentence. Our large-scale study on the entire dataset show that our method outperforms the baseline method, which is based on the exact match between the aspect signature and sentences. Our method is able to annotate 61% more sentences than the baseline method with a precision of 0.8.

The third problem we tackle is the generation of personalized, aspect-based summaries. We propose an optimization algorithm to maximize the coverage of the aspects the user is interested in and the representativeness of sentences in the summary subject to a length and similarity constraints. We propose a novel method for measuring the representativeness of sentences based on the popularity of opinions that describe the aspects.

Finally, we perform three user studies to assess the quality of the aspect-based personalized summaries we generate with real users. The results of the user study

[1] https://www.tripadvisor.com/
[2] https://www.imdb.com/

show that the approach we propose extracts less redundant and more representative summaries compared to the state of art method for generating summaries.

Acknowledgements

I would like to express my heart-felt gratitude and appreciation to my advisor Renata Teixeira for the continuous support and assistance of my Ph.D., for her guidance, and enthusiastic encouragement. I am also thankful for the inspiration and for the excellent example she has provided as a successful woman in computer science. I wish to present my special thanks to my advisors Vassilis Christophides and Christophe Diot for their encouragement, patience, and vast knowledge, and for always challenging me and helping me evolve my ideas. I would also like to thank my family: my mom, dad, sister and nephew, and friends for their moral and emotional support throughout this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Online reviewing has flourished in the recent past as users help each other with information about products they have bought, places they have been to, or content they have seen. Users provide reviews about practically everything available online or offline: hotels, restaurants, books, movies, etc. Recent studies show that 91% of people regularly or occasionally read online reviews, 84% trust online reviews as much as a personal recommendation, and 93% say that online reviews impact their purchasing decisions.[1] In this dissertation, we focus on online reviews of movies. Movie review mining is considered more challenging than mining other types of reviews, such as product reviews [106]. Movie review sentences are more complex (e.g., covering several aspects in one sentence) due to the nature of movies as products of the creative industry. A number of crowed-sourced websites collect reviews of movies, for example, IMDB, Amazon, and Rotten tomatoes. These sites often have hundreds of reviews for one single movie. Such large number of reviews is crucial to ensure the system contains sufficient information about each movie and to increase people's trust in the system, but unfortunately it also makes it impractical for anyone to process all the available information.

User reviews often contain two forms of information: a numerical rating (usually ranging from 1 to 10 or a 1 to 5 star rating) and free-form textual comments about the movie. Websites summarize ratings of individual reviewers with an average rating and a rating distribution per movie. These statistics give high-level indication of the reviewers' collective impression of a movie, which can help distinguish really poor from great movies, but most movies have good ratings. On the other extreme, understanding the reviewers' opinion of the movie from textual reviews requires reading a huge amount of information, which is tedious and time consuming for most people; especially if users want to compare between different movies to decide what to watch. Take the example of two popular animation movies: "Monsters Inc." and "Shrek". Both movies have almost the same rating on IMDB (8.1 vs. 7.9). A detailed comparison of the two movies, however, would require going through over a thousand reviews as Monsters Inc. has 620 reviews and Shrek 991. Instead of presenting the too coarse rating or the too detailed reviews, this dissertation argues for presenting a summary of the reviews.

Automatic text summarization is an active area of research. Hovy and Li [35] define text summarization as "a text produced from one or more texts that contains the same information as the original text and is no longer than half of the original text". Mani et al. [63] define the goal of summarization as a process of finding the source of information, extracting content from it and presenting the most important content to the user in a concise form and in a manner sensitive to the needs of user's application. Traditional single document summarization produces a short text from a

[1] See: https://www.brightlocal.com/learn/local-consumer-review-survey/ and http://learn.podium.com/rs/841-BRM-380/images/2017-SOOR-Infographic.jpg.

long text by extracting some "important" sentences, whereas multi-document summarization finds differences among documents and discards repeated information.

A specific branch of text summarization focuses on online reviews [57, 54, 55, 37, 106]. This area focuses on summarizing opinions from customer reviews or comments on social networks. This problem differs from a standard document summarization for two reasons. First, reviews often express an opinion about a product or about aspects of the product. *Aspects* also called features describe the characteristics of a product that users give opinions about, for example, "Story" and "Visual effects" for movies, or "Ambiance" and "Service" for restaurants. The problem is that most reviewing websites do not offer users the opportunity to review or rate the different aspects of the product. So the goal of review summarization is then to capture the main opinions about aspects of the product. Second, the "importance" of a sentence in traditional text summarization is often defined based on particular knowledge of how texts are structured. For example, in traditional text summarization a common assumption is that sentences that contain words appearing in the title and headings of a document are more important than other sentences [31]. In opinion summarization, on the other hand, such structural knowledge is not helpful as the summary must capture the main opinion (or sentiment) about each relevant aspect of a product. In this dissertation, we refer to this area of opinion summarization as aspect-based summarization as the goal is to summarize the opinion for each aspect of a product.

Existing work on aspect-based summary generation, however, focuses on extracting information that summarizes the opinions about aspects in general, ignoring the personal preferences of the user. Users tend to have different interests regarding the aspects of a product [5]. For example, a user might be more interested in the actor performance, while another user might be interested in the story of the movie. Although personalization of summaries has received some attention in the context of general text [101, 73, 33], there has been no prior work that brings personalization to aspect-based summarization of movie reviews. This dissertation develops novel methods and algorithms that together generate personalized, aspect-based summaries of a given set of movie reviews.

1.1 Text summarization background

Text summarization has been an active area of research for more than half a century. In late 50s, Luhn [60] created abstracts of documents automatically. Over decades there have been many summarization systems dealing with different forms of summarization. Existing summarization systems involve identifying the most important (i.e. relevant) sentences from the text and assembling them in a concise summary [89]. Based on some characteristics that determine the importance of sentences such as the popularity of the words contained in a sentence, an initial summary is generated with the most important sentences.

We can divide techniques of summarization into two categories: abstractive [39] and extractive summarization [28, 88] of a single or multiple documents. Extractive summarization aims to choose parts of the original document such as sentence part, whole sentence or paragraph. Abstractive summarization paraphrases content of the original document with respect to cohesion and conciseness of the output summary. Even though summaries created by humans are usually abstractive, most of the summarization research today has focused on extractive summarization. Purely

extractive summaries often times give better results compared to automatic abstractive summaries. In fact abstractive summarization requires linguistic knowledge to perform a deep analysis of the textual content to solve problems like inference and natural language generation. Theses techniques are relatively harder than data-driven approaches such as sentence extraction, which means that extractive summarization limits the risk of forming grammatically incorrect sentences. Furthermore, most existing abstractive summarizers often rely on an extractive preprocessing component to produce the abstract of the text [45, 42, 100]. Consequently, *this dissertation focuses on an extractive summarization approach*.

With the rapid growth of the web in recent years, we can now find reviews almost everywhere—blogs, social networks, news portals, e-commerce sites, etc. This type of content led to a new emerging type of summaries, aspect-based summaries [57, 54, 55, 37, 106], that take into consideration the opinion a user expresses of an aspect of a movie, product, place, etc. Aspect-based summarization is usually decomposed in three tasks: (1) identifying the aspect terms and the opinion attached to each aspect in a sentence; (2) determining the class of the aspect term and the polarity of the opinion word; and finally (3) producing a summary using the discovered information. This dissertation makes contributions in these three tasks.

1.2 Personalization

One answer to help users cope with the continuous increase in the amount of information available online is personalization, or selecting the most relevant information for each user. Users tend to prefer when systems present tailored content that is most appealing and interesting to them [73]. Personalization aims to customize the results based on the user's explicit or implicit interests and desires [74]. Nielsen Media Research[2] showed that different TV shows appeal to different demographics of users. Another study [] discovered that gender, age, culture, and social influences play a role in the kinds of products people like. Personalization is widely adopted today with a large number of websites that recommend music, movies, and restaurants among other products based on a user profile.

Personalization has been explored in various domains, including in the last years to generate personalized summaries. A good summary is not just a compact version of the original text, it should also reflect user's preferences and interests [5]. For example, Banos et al. [3] use a machine learning classification framework to filter news following the user's choices. Campana and Tombros [12] present a new method to generate personalized summaries of blog posts. They build the user model by updating and analyzing the interaction of the user with the blogging application. Their evaluation showed that 88% of users found that personalized summaries contained more relevant information about the articles compared to 75% for the general summary. Despite these results, the user model relies on an intrinsic representation of user interests via keywords, annotations or document categories, and not on the aspects. This user model prevents us from applying their solution to summarizing movie reviews.

Berkovsky et al. [5] present a new approach to solve the problem of aspect-based personalized summaries. The paper presents a preliminary evaluation of users' attitudes towards personalized aspect-based text summarization. The domain of this study is natural sciences, and the text input for the method is Wikipedia documents. They model the user profile as a vector of user interest on aspects. Their summary

[2] http://www.nielsenmedia.com/

includes the first m most relevant sentences for a given aspect and the amount of text for a given aspect is proportional to the user's interest in it. They show that users (1) prefer personalized summaries that reflect their interests over general ones, (2) have a preferred summary length (not too long or too short), and (3) consider the faithfulness to the original document of personalized and general summaries to be almost the same. Apart from the fact that this approach is not designed for text reviews, and does not take the opinions into consideration, their conclusions motivate our work as users preferred the personalized summaries.

To the best of our knowledge, no prior work has addressed the problem of personalized, aspect-based summarization of reviews in general and of movie reviews in particular. We believe that the personalization of aspect-based summaries will enable a potential user to easily see how reviewers feel about aspects of the movie relevant to her.

1.3 Contributions

In this dissertation, we propose methods and algorithms that together generate personalized, aspect-based summaries from movie reviews. We divide this problem in three main sub-problems: (i) an algorithm that extracts what "aspects" of a movie are discussed in reviews, (ii) a method for annotating sentences with aspects, (iii) an optimization algorithm that generates aspect-based summaries, personalized for a particular user. Finally we perform three user studies to assess the potential utility of the aspect-based personalized summaries with real users. Our user studies results show that the proposed method outperforms the baseline method that generates generic summaries i.e. 64% of users preferred our summary compared to 36% for the baseline summary.

1.3.1 Movie Aspect Signature Detection

The problem of aspect signature detection aims at developing an algorithm to identify the set of terms that reviewers use in the set of reviews to talk about each aspect. We rely on aspect signatures to identify which aspects are discussed in each sentence. Extracting aspect signature is challenging because users tend to use different words to describe the same movie aspect. For example, "Fantastic VFX" and "Fantastic special-effects" are identical sentences even if users employ different words. Our goal is to find all terms describing a given aspect in the aspect signature to identify all sentences talking about an aspect and to avoid redundant sentences in the final summary.

One trivial solution to identify aspect signatures would be using WordNet [72] or an already available thesaurus in order to find the related words and to group them. The first problem with dictionaries is that they do not contain domain specific knowledge, for which a domain corpus is needed. For example using a dictionary to find the synonyms of the word "picture" will result in the following non exhaustive list "painting","drawing", "sketch" while in the movie domain "picture" is synonymous to "movie". The second problem is that a dictionary may miss some synonymous words, for example, using a dictionary we cannot detect that "animatronics" and "SFX" are synonymous words for "special-effect".

The first contribution of this dissertation is a new algorithm that extracts aspect signatures from the corpus of reviews. We propose a clustering algorithm that extracts aspect terms and groups them into aspects for movie reviews. Our work differs from prior studies in three ways. First, it identifies aspect terms and group them simultaneously. Existing clustering-based solutions [93, 58, 2] take a two-step approach that first identifies features and then employs standard clustering algorithms (e.g., k-means) to group aspects into groups. We propose to combine these two steps into a single clustering process, in which different words describing the same aspect can be automatically grouped into one group. Second, most related work use first a filtering step based on the frequency to identify aspect terms, our method on the other hand is able to extract even unpopular terms in the corpus that are mainly compound terms or abbreviations. Third, our method requires no hand-crafted patterns or a human labeling efforts, it only requires a set of seed terms.

Our approach leverages a word embedding model, namely Word2Vec, which was introduced by Mikolov et al. [70]. When trained on a corpus of documents, Word2Vec learns for each word its representation in a vector space. Words that are used and occur in the same contexts in the corpus tend to have similar meaning and are mapped to nearby points in this vector space. Word2Vec model is completely unsupervised and the similarity between words is dependent on the context. In other words, Word2Vec identifies lexical similarities in a specific context. It will capture synonyms based on this context, but also similar words, i.e., words that are not synonyms but are used in the same context with a close semantic (e.g. father and mother).

The main advantage of our approach is that its extensible, which means that if we have more data we can still enrich the vocabulary that we find in the aspect signature. Second the method is fairly simple as it makes use of no syntactic dependency parsing nor NLP based heuristic.

1.3.2 Annotation of sentences with aspects

Before generating aspect-based summaries, we must annotate each sentence in the reviews with the aspect(s) discussed in the sentence. Note that reviewers can comment on multiple aspects in a single sentence [37]. For example the sentence "Talented cast, and mind-bending visual-effects" covers both "Cast" and "Visual-effects", whereas "Very funny comedy movie, couldn't stop laughing" covers only "Comedy". Most prior work on annotating sentences with aspects has considered that a sentence is discussing an aspect if the sentence contains the aspect term [98, 37, 81, 39, 38], in other words, they only consider an exact match. We argue that the exact match is too restrictive in our case, because when writing free text movie reviews, reviewers may use other terms that are similar to the terms in the aspect signatures, even if not explicitly included in the list.

Our second contribution is a novel method to annotate sentences with aspects that considers a similarity distance between the aspect signature and sentences rather than the exact match. This method allows us to annotate more sentences with aspects, which is crucial to obtain a rich set of sentences that covers the range of opinions about each aspect, to build our summaries. Specifically, our method is based on the similarity between the aspect signatures and the noun terms in the sentence.

One challenge is how to evaluate such methods as there is no existing dataset of sentences extracted from movie reviews annotated with movie aspects. We conduct a user study to first tune the aspect signature algorithm and the sentence annotation parameters, and then to compare our sentence annotation method with the existing

methods. We use the Prolific crowd-sourcing platform, where three workers annotate 600 different sentences with aspects. As a contribution of this dissertation to research in sentence annotation with aspects, we make this dataset publicly available.[3]. Our method has a precision of 0.8 and a recall of 0.6. Our large scale evaluation shows that our annotation method based on similarity is able to annotate 61.5% more sentences than using the exact match method.

1.3.3 Aspect-based, personalized summary generation

The main goal of this dissertation is to generate personalized, aspect-based summaries for movie reviews. We are given a set of opinionated sentences annotated with aspects, which have different levels of popularity in the original set of reviews, and user profiles expressed as whether the user is interested or not in each aspect. As justified in Section 1.1, we opt for an extractive approach, so the problem we solve is to select a subset of these sentences for each user profile given a summary size constraint.

One challenge is that users may express the same opinion about one aspect using different terms, for example, users may describe the aspect "Story" using terms such as "superb", "excellent", and "awesome", which all convey the information that the story of the movie is excellent. A naive approach that fails to recognize that these terms are similar may incur in two issues. First, the final summary may contain multiple redundant sentences. Second, the final summary may fail to contain the most representative opinions about the aspect. For example, if six sentences say that the story is poor, five express that the story is excellent, other five that it is awesome, and other five that it is superb, then the method may mistakenly consider that the most representative opinion is that the story is poor. We introduce a novel method to group similar opinions about aspects, which allow us to accurately compute the representativeness of opinions about each aspect of a movie and to avoid redundancy in the final summary.

The second challenge we face is how to combine personalization and representativeness to generate the final summary. We formulate the personalized, aspect-based summary generation problem as a multi-objective optimization problem. Our objective is to maximize the coverage of the aspects in the user profile in the final summary and the representativeness of sentences in the summary subject to a length constraint.

The final challenge for any summary generation work is evaluation. In particular, in our case we generate summaries from sometimes hundreds of reviews, which are often each relatively long. It is hence impractical to request human workers to read all reviews to judge the quality of our summaries. Instead, we build a two-step evaluation method. One experiment that focuses on the quality of our function to compute sentence representativeness, and the other to evaluate the quality of our summaries. Our results show that workers prefer our summaries over the baseline summaries.

1.4 Dissertation Outline

The rest of this dissertation is structured as follows. Chapter 2 represents related work on aspect signature extraction, annotation of sentences with aspect, as well

[3]The dataset is available at: https://drive.google.com/file/d/1gTkATbbO1mtngCxPTnQA7evhcxn-9cBC/view?usp=sharing

as summarization in general text, personalized and aspect-based summarization. Chapter 3 presents movie review datasets we use to develop and evaluate our methods together with the methods we use to compute similarity between words and sentences. Chapter 4 presents our method to extract signatures of aspects. Chapter 5 shows the algorithm to annotate sentences with aspects using the aspect signatures we extract. Chapter 6 takes as input the review sentences annotated with aspects using the methods from Chapters 4 and 5 to develop our optimization algorithm to generate summaries of movie reviews according to the user interest in aspects. We conclude this dissertation with a summary of the main contributions and directions for future research in this area in Chapter 7.

Chapter 2

Related Work

In this dissertation, we design a system that takes movie reviews and user preferences (expressed in terms of movie aspects) as input, and creates a personalized summary using extractive methods [5].

In this chapter, we survey techniques related to the three building blocks of our movie review summarization system. In Section 2.1 we present techniques for extracting aspect signatures from unstructured textual reviews. Section 2.2 describes how existing methods annotate sentences with a set of aspect signatures previously extracted. Finally, in Section 2.3 we present multi-document summarization using extractive methods and focus on related work in personalized summarization of opinionated text in Section 2.4.

2.1 Aspect signature extraction

The first step of our summarization method is to extract from reviews terms that are related to aspect seed terms. In movie reviews, people often use a large vocabulary to describe movies aspects (i.e. genres). For example,"Special-effects" and "VFX" refer to the same "Visual-effects" aspect. Extracting these signatures is critical for generating a summary.

2.1.1 Mapping aspects to the text

In this section, we survey methods that rely on a domain-specific knowledge of aspects and attempt to map them to the textual content of a document corpus.

Lexicon-based

Lexicon-based methods collect a set of terms for each aspect in an iterative way. Starting from a seed term for each aspect, they are searching for synonyms and antonyms of these terms in dictionaries or thesauri like WordNet [72]. The newly found words are added to the aspect signatures then the next iteration starts. This iterative process stops when no new words are found. After completion of this process, the list is checked manually to remove or correct errors.

Work by Blair-Goldensohn et al. [9] takes reviews from a service of interest, bootstraps a lexicon with WordNet for opinion detection, and updates the polarity scores for each word to find out how positive or negative an opinion is. Then it makes summarization related to the aspects.

Samha et al. [90] use aspect information provided by the manufacturer to build a manual list of aspects. This list of aspects is used to identify similar aspects from reviews with the help of WordNet. To identify opinion words, opinion lexicon was used which searched for the most common used opinion terms.

The effectiveness of these methods is constrained by the specificity of terms found in the employed linguistic resources. Although WordNet and other thesaurus dictionaries can help to some extent, they are far from sufficient because many synonyms are domain dependent [55]. For example using WordNet, the term “picture” is found to be semantically related to related {“painting", “drawing", “sketch"} and not to “movie”. Furthermore, domain specific jargon used in movie reviews cannot be detected. To overcome this problem, our algorithm uses our *corpus-based* approach. For example, our algorithm is able to detect that “animatronics” and “SFX” are synonymous words for “special-effect". Many aspect expressions are bigrams, which cannot be easily handled with dictionaries. We choose to pre-process our corpus in order to identify multi-word phrases, and we rely on a word embedding trained with the corpus of movie reviews, which captures similarities specific to the movie domain.

Rule-based

Bagheri et al. [1] use a bootstrapping algorithm approach, which needs an initial set of seed of aspects. First they propose a method to learn multi-word aspects. Second they define several POS patterns based on heuristics to detect the aspects. Third they define a new metric which uses inter-relation information among words, and aspect frequency. This metric is used through a bootstrapping algorithm to generate the final list of aspects. Finally two pruning methods based on the relations between aspects in reviews are presented to remove incorrect aspects.

Wang et al. [97] use context-dependency property to learn product aspects and opinions simultaneously. They use a bootstrapping method to identify product aspects and opinions which takes lexicon of seeded opinion words as input to extract product aspects. Further, they use the pointwise mutual information [13] to identify association among aspects and opinion words.

For instance Qui et al. [83] propose an approach that uses opinion words as seed terms for example “good” or “bad”. It utilizes these opinion words to find new opinion words and aspects. The method uses several syntactic relations that link opinion words and aspect. The newly extracted opinion words and aspects are used to extract more opinion words and aspects in the same way. The process continues until no additional opinion words can be added. Zhang et al. [104] improves this method by adding additional rules. They introduce “part-whole” and “no” patterns to identify aspects. In any review, an aspect could be the part of the object class. For example in the sentence “the engine of the car”, the engine is part of the car. This is identified by part-whole pattern. They also study that users express their views in phrases rather than in complete sentence like “no noise”. To handle these kinds of aspects the “no” pattern is used. They also study the importance of different aspects in the same product and ranks the aspects with the help of aspect relevance and aspect frequency.

Zhuang et al. [106] start with specific opinion words and take their synonyms and antonyms from WordNet [72]. Then, they find the aspect-opinion pairs using grammatical dependency and summarize the results. This work is specific to the movie domain.

Another more recent work [105] propose a more sophisticated method that uses two inputs to produce the final aspect clusters: aspects taxonomies of products and the product publicly available reviews. This method uses a set of distance metrics combined with an optimization strategy.

These method have two steps. First, aspects are extracted based on association rules or dependency patterns. In the second step, aspects are grouped into aspects using clustering algorithms. In contrast, our method extracts terms and groups them based on the aspects they are related to in one step. Also in most of the cases when free format reviews are considered, the POS taggers do not function at the expected level as grammar is not guaranteed in user generated text. On the other hand, our method does not rely on any lexical nor POS rules to detect aspect terms, instead, it relies on the similarity between the terms in the movie domain.

2.1.2 Mining aspects from the text

Three main approaches are used in the literature to extract aspects from reviews: information retrieval approaches, supervised and unsupervised approaches. We discuss each line of work below.

Information Retrieval Approaches: Frequency and lexicon-based

In reviews, people are more likely to give their opinion about aspects, which suggests that aspects should be frequent nouns or *noun phrases* such as "motion picture" or "plot line". However, not all of the frequent nouns are aspects. Therefore, different filtering techniques [37, 81, 39, 38] can be applied on frequent nouns to filter out non-aspects. The main advantage of these methods is their simplicity.

Pioneer work by Hu et al. [38] uses a data mining algorithm. Nouns and noun phrases are identified by a part-of-speech (POS) tagger. Their occurrence frequencies are counted, and only the frequent ones are kept. A frequency threshold can be chosen experimentally. This approach works because the vocabulary used is quite similar among reviews. Nouns are frequently used to describe relevant and important aspects. Hence, infrequent nouns are likely to be unrelated to aspects or less important. Although this method is very simple, it is actually quite effective.

The precision of this algorithm is improved in [81]. It tries to remove noun phrases that do not describe aspects. Each noun phrase is scored by computing a pointwise mutual information (PMI) [13] score between the noun phrase and some *meronymy* discriminators associated with an entity class, e.g., a "Camera" class. Meronymy is defined as a relationship between words that mean "part of" [72] . For example meronymy discriminators for the "Camera" class are, "of camera," "camera has," "camera comes with,".

Long et al. [57] extract aspects (nouns) based on frequency and information distance. Their method finds first the aspect words using the frequency-based method [38]. Then it uses the information distance defined by Rudi et al. [14] to find other related words to an aspect, e.g., for aspect "Price", it may find "$" and "dollars". All these words are then used to select reviews which discuss a particular aspect.

Yan et al. [102] proposed a PageRank algorithm [77] to find the association among aspects and sentiment words. They also use the synonym lexicon to expand the aspect list and used the same lexicon, along with the association among explicit aspect and opinion word, to identify the implicit aspects.

Many companies are using these techniques for analyzing user feedback on their products or services [54]. However, these methods tend to produce too many terms that are not related to aspects and miss low-frequency aspect terms. Our algorithm to extract signatures of aspects overcomes both these problems. Our method is based on the similarity between a seed term and our corpus. We use a similarity threshold to ensure that the terms we include in the signatures are related to aspects. In

Chapter 4 we show how our method is able to extract also low-frequency aspect terms.

Unsupervised Approaches: Topic models

A different line of work on aspect signature extraction is based on topic models. There are two main basic models, pLSA (Probabilistic (Probabilistic Latent Semantic Analysis) [34] and Latent Dirichlet Allocation (LDA) [10]. The goal of topic modeling is to discover the topics in a given corpus. The output of a topic modeling approach is the topic-specific vocabulary distributions, the document specific word assignments and the document specific topic proportions.

Mei et al. [67] propose an aspect-sentiment mixture model for sentiment analysis that they call topics sentiment mixture model (TSM). They assume that a review is generated by sampling words from a set of topic distributions and two sentiment distributions which correspond to positive and negative, respectively. Specifically, they build a joint model, that is based on an aspect (topic) model, a positive sentiment model, and a negative sentiment model learned with the help of labeled training data. Their model is based on pLSA. Most other models proposed by researchers are based on LDA.

Branavan et al. [11] use *Pros* and *Cons* keyphrases that reviewers annotate in their reviews i.e. Pros: "innovative story" or Cons: "Cheap cgi", to find aspects in the detailed review text. The model has two parts. The first part clusters the keyphrases in Pros and Cons based on their distributional and lexical properties. The second part builds a topic model that models aspects in the review text. The keyphrase clusters and document topics are modeled jointly in the final model. The two parts are integrated based on the idea that the model biases the assignment of hidden topics in the review text to be similar to the topics represented by the keyphrases in Pros and Cons of the review, but it also permits some words in the document to be drawn from other topics not represented by the keyphrases. This flexibility in the coupling allows the model to learn effectively in the presence of incomplete keyphrases, while still encouraging, this approach still does not separate aspects and sentiments.

Titov et al [94], shows that global topic models such as LDA might not be suitable for detecting aspects. The reason is that LDA depends on topic distribution differences and word co-occurrences among documents to identify topics and word probability distribution in each topic. However, opinion documents such as reviews about a particular type of products or services are quite homogenous, meaning that every document talks about the same aspects, which makes global topic models ineffective and are only effective for discovering general topics i.e., James Bond movies if applied on a movie collection. The authors then propose the multigrain topic models. The global model discovers global topics while the local model discovers aspects using a few sentences (or a sliding text window) as a document. Here, each discovered aspect is a unigram language model, i.e., a multinomial distribution over words. Different words expressing the same or related facets are automatically grouped together under the same aspect. This technique as well does not separate aspects and sentiment words.

Lu et al. [58] presents a method that models the co-occurrence of aspects at the level of the modifiers (opinion terms) they use. For example if two aspect terms tend to co-occur with each other (such as, "ship" and "delivery" co-occurring with the modifier "fast" in "fast ship and delivery") then both terms should have a high probability to belong to the same aspect topic. The drawback of this method is that

extracts many non aspect terms, i.e. { recommended, was } and { buy, do} are considered respectively the top two terms with the highest probability to belong to two different aspect topics . This approach, is also only tested on eBay comments, which are rather short on the apposite of movie reviews that are more verbose.

While these LDA-based approaches provide an elegant and unsupervised model, they produce topics that are often not directly interpretable as aspects, and thus require manual labelling to achieve a readable output. Another issue is that topic modeling finds very general and frequent topics or aspects from a large document collection, but it does not find aspects that are locally frequent but globally not so frequent. Such locally frequent aspects (i.e. story) are also useful in the use case of personalized aspect-based summarization, as they might be very relevant for a specific user. Those very general and frequent aspects can also be easily found by the methods discussed earlier. In short, the results from current topic modeling methods are usually not granular or specific enough for many practical aspect-based applications. However, topic modeling can be useful to identify what a document collection is about.

Supervised Approaches: HMM and CRF

Supervised techniques need manually labeled data for training. That means that aspects and non-aspects terms should be manually labeled in the corpus. Many algorithms based on supervised learning have been proposed in the past for aspect extraction. The current state-of-the-art supervised learning for aspect extraction are Hidden Markov Models (HMM) [84] and Conditional Random Fields (CRF) [51].

For example, Li et al. [53] propose a Skip-chain CRFs and Tree CRFs for the extraction of aspects that are based on Conditional Random Fields (CRF). First they use linear-chain CRFs to identify the sequential dependencies among contiguous words. They learn that if two words or phrases are connected by conjunction “and”, then both the words have the same polarity and if they are connected by “but”, then both have the opposite polarity. To overcome long distance dependency they use Skip-chain CRFs to find aspects and opinions. Also, they propose to use Tree CRFs to learn the synthetic structure of the sentences in the reviews. Skip-chain CRFs provide the semantic relations with respect to conjunctions and Tree CRFs provide dependency relations among different words in the sentence. Further, they propose Skip-Tree CRFs to combine both methods described above and use these trees to extract aspects and opinions. Unlike the original CRF, which can only use word sequences in learning, Skip-CRF and Tree-CRF enable CRF to exploit structure features. Liu et al. [56] use sequential pattern rules. These rules are mined based on sequential pattern mining considering labels (or classes).

For the extraction of aspects and opinions from the customer reviews, a lexicalized HMM- based model is proposed by Jin et al. [41]. This work not only identifies the product aspects and their opinions but also identifies the sentences which contain aspect opinion pair and categorize the opinion words as negative or positive. They first describe the two tag sets, first is the basic tag set which defines different entities (opinions, aspects or background words), and the second tag set defines the patterns for these entities i.e. what is the position of a word in the aspect/opinions sentence. With the help of these tag sets they manually tag each sentence representing the patterns between aspects and opinion words. This method uses HMM along with maximum likelihood estimation (MLE) to find the appropriate sequence of hybrid tags (manual tags and actual tagged data) that maximize the conditional probability.

Kobayashi et al. [46] proposes another supervised dictionary-based method to extract aspects from blogs. The method first finds candidate aspect and opinion word pairs using a dependency tree, and then employs a tree-structured classification method to learn and to classify the candidate pairs as being an aspect opinion pairs or not. Aspects are extracted from the highest scored pairs. The features used in learning include contextual clues and statistical co-occurrence clues.

Using supervised algorithms to solve the problem of aspect extraction achieves good results in terms of precision and recall, however it presents several drawbacks. First, training a supervised algorithm requires a large amount of manually labeled data, which is not available in most cases. Second, supervised algorithms introduce the problem of domain adaptation, which is when an algorithm is trained on a labeled data from a domain S it should also perform well on data from another domain T [17, 40]. This adds another layer of difficulty which is adapting the size of the training data and selecting the best features that minimize labeling errors for different domains. Our algorithm on the other hand does not require prior knowledge nor labeled data in the corpus to extract aspects, it only needs a set of aspect seed terms as input.

2.2 Annotation of Sentences With Aspects

The second core components of our proposed summarization method, is to understand the aspects discussed in a sentence.

There is very little related work in this area. Most research consider that a sentence discusses an aspect only if there is an exact match between the sentence and the aspect signature [98, 37, 81, 39, 38]. We consider this method i.e. *exact-match* as our first baseline method. This approach annotates a sentence with an aspect only if a term from the aspect signature is explicitly mentioned in the sentence. The second baseline method we consider is the *count of matching terms* [98]. This method annotates each sentence with the aspect that shares the maximum number of terms with the aspect signature. The sentence is annotated by multiple aspects in case of ties. To the best of our knowledge, there is no work that considers a similarity distance between the aspect signature and the sentence. We argue that the exact match and the count of matching terms is too restrictive, because reviewers may use other terms that are similar to the terms in signatures, even if not similar to the aspect seed term. One of our contributions is to solve this problem. Chapter 5 presents the generic sentence annotation method we use, that covers different review writing styles. Specifically we conceive an adequate similarity metric between the review sentences and the aspect signatures. The large scale evaluation we conduct in Chapter 5 shows that our annotation method based on similarity is able to annotate 61.5% more sentences than using the baselines method with a precision of 0.8.

2.3 Text summarization

Automatic text summarization has been studied for a long time due to the need of handling large amount of electronic text data [65, 89]. There are two representative types of automatic summarization methods. Extractive summarization selects representative text segments, usually sentences, from the original documents [35]. Abstractive summarisation does not directly reuse the existing sentences from the input data; instead it analyzes documents and directly generates new sentences [29]. Our work is related to extractive summarization. It is hard to generate readable,

coherent, and complete sentences, studies on extractive summary are more popular than those on abstractive summary. Research in the area of summarizing documents has focused on extracting salient sentences from text and coherently organizing them to build a summary of the entire text [35, 75, 50]. In the following subsections we present the main techniques for summarizing single or multiple documents using extractive methods. Then, we focus on summarization of opinionated text for aspects-based summarization.

2.3.1 Single document Vs Multi-document summarization

The goal of single document summarization is to generate an abstract summary for a single document. On the other hand the goal of multi-document summarization is to aggregate information across different documents that are related to the same topic but with different perspectives [85]. Multi-document summarization is more challenging than single document summarization. The first challenge is the coherence of the summary. Coherence measures how much it's easy or fluent to understand a summary. As pointed out in [86, 4, 30], it is difficult for multidocument summarizers to produce coherent summaries, since it is less straightforward to rely on the order of sentences in the underlying documents than in the case of single-document summarization. The second challenge is that the summary should extract representative sentences from multiple documents, which is not the case of single document summarization [86, 99].

The vast majority of extractive methods for single and multi-document summarization identify which sentences are important using unsupervised or supervised learning techniques. Early extractive techniques are based on simple statistical analysis about sentence position, term frequency [25, 60], or basic information retrieval techniques such as inverse document frequency [92].

For supervised methods, summarization is often regarded as a classification task or a sequence labeling task at sentence level, and many supervised learning algorithms have been investigated including Hidden Markov Models [84] and Support Vector Regression [23]. However, such a supervised learning paradigm often requires a large amount of manually labeled data, which are not available in most cases.

The unsupervised methods are usually based on a combination of linguistic and statistical aspects such as term frequency, sentence position, lexical chains, rhetorical structure. Clustering-based methods usually select one or more representative sentences from each topic to produce a summary with minimized redundancy and maximized coverage [76].

Graph-based methods are also very effective and are becoming more and more popular: LexRank [26] is a representative algorithm which measures the centrality of sentences, converts sentences into a graph structure, and finds central sentences based on their popularity. The graph based approach shows good performance for both single and multi-document summarization. Moreover, it does not require language-specific linguistic processing so it can also be applied to other languages [69].

In general, aspect-based summarization and multi-document text summarization present similar challenges. However, an aspect-based summary is quite different from a traditional single document or multi-document summary. An aspect-based summary concerns aspects and sentiments about them. Single document summarization produces a short text from a long text, while multi-document summarization finds differences and commonalities among documents. Neither of them

explicitly captures different aspects discussed in the document and sentiments about them, nor do they have a quantitative side.

2.3.2 Aspect-based summarization

Aspect-based summarization is one of the most popular types in opinion summarization and has been heavily explored over the last few years. It first finds aspects of the target and obtains statistics of positive and negative opinions for each aspect [37, 38, 39, 55, 59, 68, 94, 106].

Aspect-based summarization techniques have attracted a lot of attention with the growing amount of crowd-sourced reviews in the web. But due to the characteristics of the input data, aspect-based summarization differs from text summarization [37]. Sometimes, opinions are provided with additional information such as rating scores. In addition, the summary formats proposed by the majority of the aspect-based summarization literature are more structured with the segmentation by aspects and polarities (i.e. positive or negative opinions)[56]. However, text summarization techniques are useful in aspect-based summarization for text selection and generation step. After separating input data by polarities and aspects, classic text summarization techniques can be used to generate the most representative text snippets from each category [54].

There are several surveys that analyze the existing work [44, 78]. General aspect-based summarization focuses on finding aspects among articles and clustering positive and negative opinions on those aspects. Sentences are then selected according to one or multiple criteria to generate the final summary. Most of the results of aspect-based summarization focus on showing statistics of the number of positive and negative opinions [38, 39, 106]. Statistical summary directly uses the processed results from the previous two steps - a list of aspects and results of sentiment prediction. By showing the number of positive and negative opinions for each aspect, readers can easily understand the general sentiments of users at large. Along with the positive and negative occurrences, all sentences with the polarity is shown. Hu et al. [39] shows statistics in a graph format. With the graph representation, we can obtain people's overall opinions about the aspect more intuitively. Liu et al. [56] present a new software, Opinion observer, which shows statistics of opinion polarity in each aspect as a sentiment bar chart. The software enables users to compare opinion statistics of several products. Each section about each aspect has sentences extracted from the review and has a link to the original review. The portions of the bar above and below the horizontal line represent how many reviews are positive and how many reviews are negative.

While statistical summaries help users understand the overall idea of people's opinion, reading actual text is necessary to understand the specific opinions of users. Due to the large volume of opinions on one topic, showing a complete list of sentences is not very useful. To solve this problem, many studies [94, 68, 48, 58, 82] select representative sentences from the text and present them as a summary. Popescu et al [82] rank opinion words associated to aspects using the point mutual information score, and show the strongest opinionated word for each aspect. A sentence level summary can provide a deeper level of understanding of an aspect. Mei et al. [68] score the probability of each sentence to each topic using word probability in topic modeling of TSM model. By choosing the top ranked sentence in each category, they are able to show the most representative sentence. Ku et al. [48] on the other hand, score sentences based on the TF-IDF of their words and select the most relevant and discriminative sentence to be shown as summary.

Although these sentences selection techniques are interesting, they don't consider the problem of vocabulary diversity. For example, if the aspect "Story" is described in five sentences with the opinion word "nice", in four sentences by "imaginative", and in other three sentences by the opinion term "creative", a simple frequency count would rank the sentences containing the opinion term "nice" higher as it appears five times, whereas we can see that the major opinion about the aspect "Story" is that it is creative. Not solving this problem can also lead to having redundant sentences in the summary. We propose a new method that solves this problem, it groups sentences that have the same opinion about the same aspect and select sentences from the most popular groups. Our method also ensures coverage of different opinions of the same aspect.

2.4 Personalized summarization

Personalized summarization involves both documents and users. Marcu et al. [65] show that each user may have different views on what can be considered important and interesting information and how this important information should be presented. In the following sections we present the related work for both personalized summarization in general text and personalized aspect-based summarization.

2.4.1 Personalized summarization in general text

Personalization is often found in text summarization approaches, e.g. Hennig et al. [33] describe a query-based summarizer. Sentences and queries are represented in a the latent topic space of the PLSI model combined with a language model. The sentence importance score is a linear combination of several sentence-level aspects based on the similarity of sentence and query distributions over latent topics. Sentences are then selected in a greedy manner to create a summary.

Moro et al. [73] propose a personalized text summarization method based on latent semantic analysis [18] that focuses on learning domain. Two main sources of personalization of the summarization are taken into consideration, relevant domain terms, and annotations added by users (students) in the form of highlights or tags. Although this model presented takes into consideration the sentences highlighted by students, the evaluation does not take into consideration the personalization aspect, which means that all students are given the same text summary. Their user study results, however, suggest that personalizing summarization using students' annotations would make their results significantly better.

Park et al. [79] summarize the comments (descriptions) and tags which users add when they create a bookmark using social bookmarking service such as *Delicious.com*. The advantage of this approach is that it can summarize documents with no or minimum text, but with other multimedia content. On the other hand, it depends on the number of bookmarks and it is unable to summarize documents with no bookmarks. Another paper [103] also builds the user model using the keywords in the user annotations in the form of highlights by extending the popular TF-IDF method. Here the set of keywords contained in the annotated sentences are considered as an explicit representation of the user interest. The results of this study suggest that this approach is useful for personalization of summarization.

Another interesting approach presents a multi-tier model [22] that contains topical interests (long-term) as well as a relevance feedback tier, which takes into account the changes given by users' feedback (a short term model). In this approach, the

summary is generated by ranking sentences according to the linear function over a set of cosine similarities between each vector's interests and document vectors and selecting the top 20% of ranked sentences. Another similar example of a multi-tier model of interests is described in [21]. Their results conclude that generating a summary directed to a specific user based on a user's interests, achieved encouraging results. Their evaluation shows that human users preferred the personalized summaries over a generic summary [22].

Kumar et al. [49] generates personalized summaries based on the area of expertise and personal interests of a user. A user background model is developed using the information found on the Internet with regard to a person, such as his/her personal Web page, blog or on-line publications. Once the user profile is identified, the relevance of document's sentences is determined according to this profile. Two scoring functions, one for generic information and one for user specific are proposed. The first one relies on term frequency to extract the most relevant generic sentences, whereas the second one computes the probability of the generic sentences to contain also user specific information. In the summary generation stage, the top ranked sentences are selected and extracted. Although this approach is simple to replicate, it is not robust to multiple people sharing the same name as the algorithm cannot disambiguate the different identities. A similar approach is presented by Campana et al. [12]. It works in three steps, the first step consists in creating the user model from the sentences of the documents recently read by a user. The user model is represented as a complete graph. Each node represents a sentence, and an edge connecting two sentences represents the strength of the similarity between the nodes. A summary is constructed from the sentences which are the most similar to the most representative sentences of the user model. Users can extend the initial summary if they find it not sufficient, and the paper proposes a method to add sentences that express novel information.

While conceptually our problem is similar to personalized summarization in general text, there are some key differences that makes our approach unique. In personalized summarization in general text, a sentence is considered relevant to a user only if it's either similar to the sentences he has already read(highlighted) [12, 103, 73] or written [49], or if it's similar to the user bookmarks or annotations [79, 73]. Second the nature of text to be summarized is different, text reviews have the specificity of having aspects and opinions attached to these aspects, which means that we should ensure that the most representative opinions are covered in the final summary. Instead, we implement a summarization algorithm specifically designed for text reviews, that satisfies three important constraints (1) personalization based on the aspects that the user is interested in, (2) representativeness of opinions about aspect and (3) compactness.

2.4.2 Personalized Aspect-based summarization

There are many studies on aspect based summarization, but these studies ignore users' preferences, and their goal is to summarize the reviews for an average user [94, 68, 48, 58, 82, 38, 39, 106]. To the best of our knowledge, there are very few studies that deal with personalization of summaries. Aspect-based summarization systems usually analyze the reviews to detect the aspect-sentiment relations. Yet, this approach is not sufficient to obtain a personalized result. If a user is looking for an action movie, she wants to examine all the movies available and compare the reviews based on her personal preferences. Reading only the reviews about the aspects that

are related to her personal preferences can help make the right decision and save time.

The work of Berkovsky et al. [5] is closest to our work. The paper presents a preliminary user evaluation that assesses different aspects of users attitudes towards personalized aspect-based text summarization. The user profile is presented as a vector of domain aspects using a 4-point scale of interest in aspects: 0=no interest, 1=low, 2=moderate, and 3=high. Three experiments are suggested with the purpose of analyzing this issue. First, the user study evaluates whether the personalization of summaries has the desired effect on users or not. Then, the impact of summary lengths is analyzed. Finally, the degree of faithfulness between the personalized summaries and the original documents is assessed. The main findings of this study are very interesting. They show that (1) users prefer personalized summaries that reflects their interests over the general ones, (2) each user has a preferred summary length, and do not like too long or too short summaries, and (3) users consider the faithfulness to the original document of personalized and general summaries to be almost the same. However, this work offers only a basic way to build the personalized summaries, which is to adjust the amount of text for a given aspect to the user's interest in it. For example if the user profile is the following { Script: 1, Visual-effects:3} the summary would contain the top 1 ranked sentence about "Script" and the top 3 sentences about "Visual-effects". The main drawback of this approach is that if a user is interested in a big number of aspects the summary could be very long. Second this approach is not designed for text reviews, because the opinions are not taken into consideration. Finally the ranking of sentences according to their relevance about an aspect, is done manually.

Our work addresses these three problems: we build a complete optimization framework that combines different summary properties, i.e. compactness, representativeness of opinionated sentences, and coverage of aspects the user is interested in. In summary, none of the previous methods is designed to solve the problem of personalized aspect-based summarization, and none of them summarizes reviews while taking into consideration the aspects the user is interested in.

In Chapter 6 we compare our work to a generic summary generation method i.e. LexRank, we show that human assessors preferred the summaries we generate using our method over the summaries generated using LexRank.

2.5 Summary

Our personalized aspect-based summarization involves three steps. The first step consists in extracting terms related to each aspect. The second steps deals with sentences annotation with the corresponding aspect. The final step solves the problem of personalized aspect-based summarization. In this Chapter, we position our work with regards to each step of our method. We introduce the different methods used to generate the signatures of aspects. We notice that these methods have several drawbacks i.e. extracting too many non-aspect terms and missing low-frequency aspect terms. For this reason, we propose a new algorithm that overcome these problem that we explain in Chapter 4.

Second, we discuss the methods used to annotate sentences with aspects. Again, we notice that all methods in the literature use an exact match between the sentences and the signatures of the aspects. This lead us to propose a new similarity metric to annotate sentences with aspects that we present in Chapter 5.

We also discuss text summarization. We first present the various methods used to summarize general purpose texts and reviews. None of these methods is designed to personalize the summaries for a particular user.

Finally we present the state of the art methods to generate personalized summaries of general text. We show that none of these methods is designed to generate personalized aspect based summaries. This motivates the third contribution of this dissertation which is the design of an algorithm that summarizes movie reviews while taking into consideration the aspects the user is most interested in. We present our solution in Chapter 6. In the same Chapter, we compare our algorithm of generating personalized summaries with those from the literature (i.e. we conduct a user study to compare the performance of our techniques with the performance of the graph based method lexRank [26]).

Chapter 3

Dataset, Word Embedding, and Similarity metrics

The objective of this dissertation is to automatically generate a personalized summary of the set of reviews of a movie. It is important to validate our work on a large dataset that offers a good diversity of movies, number of reviews per movies and reviews sizes. For these reasons, we have chosen to work with IMDB,[1] a popular website for reviewing and rating movies. In this chapter, we describe first our dataset characteristics, and explain the pre-processing step we had to perform in order to maximize the relevance of our summaries (i.e. vocabulary cleaning in Section 3.2). Given the cleaned version of the reviews (which we call the *corpus*), our algorithms must understand the aspects that are discussed in each sentence and select sentences to be included in the summary. These tasks call for different types of "similarity" between words and sentences. In order to compute these similarities, we represent words from the corpus in a vector space. Section **??** justifies our choice of using a word embedding, instead of a simpler approach based on thesaurus for identifying similar words, and presents the Word2Vec model we select to create the word embedding. Finally, Section 3.3 introduces the metrics we use in order to compute the similarity between words and sentences.

3.1 Data Characteristics

The most popular movie reviewing sites are (at the time we started this work), Rotten Tomatoes,[2] Flixster,[3] or IMDB.[4] Although the users' reviews are publicly available, the sites' terms of use often forbid crawling their dataset.[5] There are a few review databases available for research for example the Amazon dataset provided by Stanford [66]. We study a large dataset with movie reviews extracted from IMDB, which is a popular website for reviewing and rating movies. This dataset is the same studied by Jmars et al. [20]. The advantage of this dataset is that it includes rating and textual reviews for a large number of movies. Table 3.1 summarizes the dataset, which was collected between June and July, 2013. It contains data of a total of 9,888 movies. We denote the set of movies as $M = \{m_1, ..., m_l\}$, where l is the total number of movies (the list of all notations can be found in Table A.2 in Appendix A). For each movie, the dataset contains meta data about the movie (e.g. movie genre, year of appearance) as well as text reviews and ratings. Figure 3.1 describes the data

[1] https://www.imdb.com/
[2] https://www.rottentomatoes.com/
[3] https://www.flixster.com/
[4] https://www.imdb.com/
[5] http://www.imdb.com/conditions/

TABLE 3.1: Dataset characteristics

#of movies	#of users	#of reviews	#of ratings	#of sentences
9,888	17,131	763,571	775,654	11,437,314

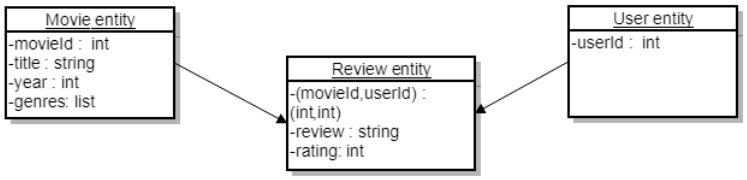


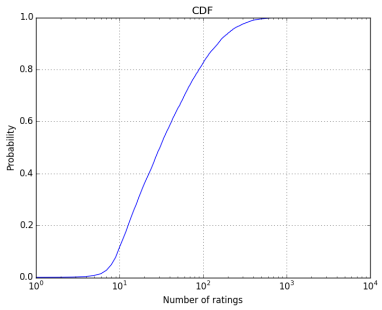
FIGURE 3.1: Dataset entities

collected in more detail. The *Movie* entity has four different attributes: the movie unique identifier *movieId*, the movie title, the year of appearance of the movie, and finally the list of the IMDB genres e.g. Romance, Comedy. The *User* entity contains the user unique identifier. Last, the *Review* entity has three attributes: the review Id, which is constructed as the (movieId, userId)-tuple, the text review, which is the actual review that the user has written about the movie, and the rating given by the user to the movie ranging from 1 to 10. Users can write a review and give a rating independently. We call $R_m = \{r_{m1}, ..., r_{mq}\}$ the set of reviews for a movie, m. This dataset contains a total of 763,571 reviews. Each review contains a set of sentences, $S_r = \{s_{r1}, ..., s_{rp}\}$.

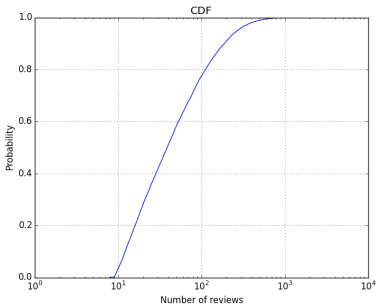
Figures 3.2a and 3.2b present the cumulative distribution of the number of ratings and the number of reviews per movie, respectively. We can observe that both distributions are skewed: The bottom 20% of movies have less than 20 reviews and ratings, whereas the top 10% have hundreds of reviews and ratings. Summarizing reviews for movies with few reviews is challenging as the data is too scarce to extract meaningful information. To account for this issue, we split the set of movies into three subsets according to the number of reviews (based on the analysis of Figure 3.2b). **Unpopular movies**, which constitute 60% of all movies, have less than 51 reviews. **Popular movies** have between 51 and 200 reviews; this class represents 30% of movies in the dataset. **Blockbuster movies** have more than 200 reviews; these represent the top 10% most popular movies. Although this particular split is arbitrary, this is not central to our algorithms, the split is only useful for us to understand how well the method works depending on the quality of the input. We evaluate the performance of our summarization algorithm for these three different movie categories, as well as for the full dataset.

Summary personalization relies on users interests in movies. We verify that our dataset has a good diversity of reviews and ratings. Figures 3.3a and 3.3b show the distribution of the number of ratings and reviews per user, respectively. Similarly, we observe a high variety of numbers of reviews and ratings across users, for example 60% of users have given less than 17 ratings and less than 10 reviews, whereas only 5% of users have given more than 120 ratings and more than 250 reviews.

Another important property of the dataset is the length of reviews. Usually natural language processing and machine learning techniques work better with longer

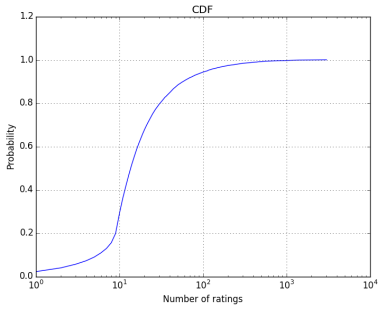


(A) Number of ratings per movie

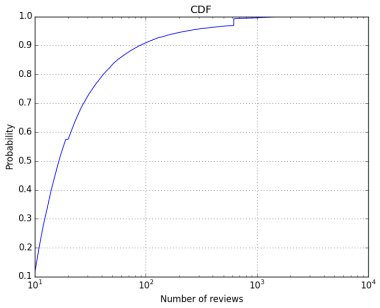


(B) Number of reviews per movie

FIGURE 3.2: Analysis of reviews/ratings per movie in IMDB dataset



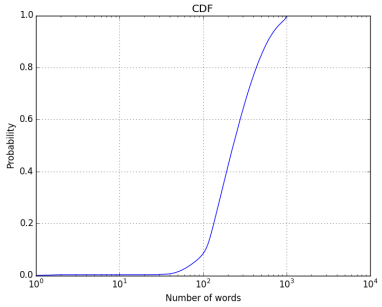
(A) Number of ratings per user



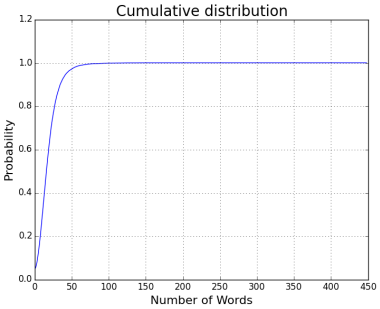
(B) Number of reviews per user

FIGURE 3.3: Analysis of reviews/ratings per user in IMDB dataset

text [70]. Fortunately, the reviews in our dataset are quite long (see in Figure 3.4a): 50% of the reviews have more than 231 words, also even if some reviews are short, we aggregate all reviews of a single movie before summarizing it for a user. Figure 3.4b presents the cumulative distribution of the number of words per sentence. 60% of the sentences contain more than 10 words. Longer sentences are more complex and consequently harder to analyze. We see that 1% of sentences have more than 50 words. We manually examined these sentences and identified that most of them are badly written with grammar and punctuation mistakes. Hence, we removed these sentences from our corpus.



(A) Distribution of the number of words of reviews.



(B) Cumulative distribution of the number of words of sentences.

FIGURE 3.4: Analysis of the number of words per review/sentence in IMDB dataset

3.2 Vocabulary cleaning

This section describes the steps we take to clean reviews from noise (e.g. typos, punctuation, html tags, and urls). The output of this cleaning step is a vocabulary that we can use to represent words as vectors. We apply standard natural language techniques to clean the dataset. We describe these techniques and justify their use below.

Tokenization Tokenization is the task of splitting a given document into small pieces, called *tokens* or *lemmas*, at the same time as removing certain characters, such as punctuation. Here is an example of tokenization. The output of sentence "This was a good movie!" is ["this", "was", "a", "good", "movie"]. We use the Penn Treebank Tokenizer implemented in the natural language NLTK for word tokenization [7, 8].

Words lemmatization Depending on the context, documents usually use different forms of the same word, for example, drive, driving, or driven. The goal of lemmatization is to reduce inflectional forms of a word to a common base form. For instance, the verb "driving" will become "drive". We use the WordNet lemmatization algorithm as implemented in NLTK [7, 8].

Collocation detection A collocation or a compound expression is a sequence of words that together have a different and unique meaning, for example, "science fiction" or "chick flick". Even if flick is a synonym of movie, it's unconventional to find "chick movie". Thus, it is important to identify these compound expressions in the whole dataset. We replace each identified collocation in the whole dataset as a single word, for example "chick flick" is systematically replaced by "chick-flick". In order to find these collocations we use a statistical metric, which is defined as follows:

$$score(w_1, w_2) = \frac{p(w_1, w_2)}{p(w_1)p(w_2)} \tag{3.1}$$

where w_1 and w_2 denote two different words in the dataset. $p(w_1, w_2)$ is the probability of the occurrence of the collocation (w_1, w_2) in the dataset and $p(w_1)$ and $p(w_2)$ are respectively the probabilities of occurrences of w_1 and w_2 in the dataset. For each collocation in our dataset we calculate its score, and we combine just the collocation that have a score higher than a threshold of 9. We select this threshold empirically as we keep only 5% of collocations having the highest scores as compound words. collocations having lower scores are not necessarily compound words for example "this-movie" and "the-film". Setting a very high threshold on collocation scores would leave many collocations not being identified.

We call the concatenation of all the reviews after applying all these steps as the *corpus*. We use this corpus of reviews to construct our vocabulary. We define *vocabulary* as the union of all words of all sentences $\forall m \in M, \forall r \in R_m, \forall s \in S_r, \forall w \in s, V = \bigcup w$.

Our algorithms needs to compute the similarity between a word that represents a movie aspect (also known as the seed word) and words found in reviews. The simplest method for establishing words similarity is using a dictionary or a thesauri. Unfortunately, this method fails to produce meaningful synonyms unless you can

find a dictionary that is specific to your review domain. The most popular dictionaries, such as Wordnet [72] are not appropriate. In Wordnet among the synonyms of the word "picture" we find the word "photograph", which is not a synonymous word of picture in the movie domain (it is rather movie or film). In order not to loose generality (we expect our algorithm to be general enough to be applied to other domains), we decided not to use the dictionary approach.

Instead, we rely on a word embedding trained with the corpus of movie reviews, which captures similarities specific to the movie domain. To compute this similarity, we first embed our vocabulary into a vector representation. The representation of a vocabulary of words into a vector space has received a lot of attention in the NLP literature as it has been shown to improve many NLP tasks, such as part-of-speech tagging [15], dependency parsing [47], and machine translation [43, 19] . In this section, we briefly discuss existing approaches for representing words in a vector space, then we present Word2Vec the approach we select to represent words as vectors in this dissertation. The word representation will create a word embedding matrix $W \in \mathbb{R}^{dim*h}$, where dim is the vector dimension and h is the size of the vocabulary. Such representation allows us to map each word w to its word vector representation $\vec{v}$.

3.2.1 Background

We represent words as vectors such that words with similar meaning are "close" in the resulting vector space. Harris' distributional hypothesis [32] states that words with similar meaning occur in similar contexts, where the context is the set of surrounding words. This hypothesis implies that the meaning of a word can be inferred from its distribution across contexts. The general idea behind representing words as vectors is to use distributional statistics to generate high-dimensional vector spaces, where a word is represented by a vector that encodes its semantic. In the following we present two main ways of capturing context, based on word counts or neural networks.

Word Count/context matrix The traditional way to represent words in a vector space is to construct a high dimensional sparse matrix, H, where each row represents a word, $w \in V$, and each column is a word context, c. The context of a word, w, is one or more words from the right and left of w. The value of each matrix cell H_{ij} measures how often a word w_i occurs with the context c_j. A popular metric of word association is "Pointwise Mutual Information" (PMI) from Church et al. [13]. PMI measures the association between two words that tells us how much more often than random the two words co-occur. Formally to compute PMI between a word w and a context word c we use the following formula [13]:

$$PMI(w, c) = log_2 \frac{p(w, c)}{p(w).p(c)} \tag{3.2}$$

$p(w, c)$ is the probability of occurrence of the bigram (w, c) in the corpus and $p(w)$ and $p(c)$ are respectively the probabilities of occurrences of w and c in the corpus. PMI values range from negative to positive infinity. Negative PMI values (which mean that words co-occur less often than we would expect by chance), however, are only reliable for large datasets. The matrix H is also ill-defined, for example a word context pair (w, c) that never co-occurred will have a value of 0, while a word context pair that rarely occurred together will have a negative value, which is inconsistent.

A more consistent approach is to use positive PMI (PPMI), in which all negative values are replaced by 0:

$$PPMI(w,c) = max(log_2 \frac{p(w,c)}{p(w).p(c)}, 0) \tag{3.3}$$

A well-known shortcoming of PMI, and also PPMI, is its bias towards infrequent events [96]. A rare context c that co-occurred with a target word w even once will often yield a relatively high PMI score because of the very small value of $p(c)$ in the PMI's denominator. This leads to having high scores for context words that rarely co-occur with w. Nevertheless, the PPMI measure is widely considered as state-of-the-art for these kinds of distributional-similarity models. Using such a high dimensional space leads to having very sparse vector representations. One way of dealing with this problem is to use a dense low-dimensional vector representation. Such vectors can be obtained by performing dimensionality reduction over the sparse high-dimensional matrix. A common method for doing so is truncated Singular Value Decomposition (SVD) [24], which finds the optimal rank k factorization. It was popularized in NLP via Latent Semantic Analysis (LSA) [18]. LSA has been proved to give better results than the aforementioned methods with relatively little text [95]. We have applied the LSA method on our corpus for the task of identifying similar words, the results were not as good as using *Word2vec* a prediction model that we explain in Section 3.2.2.

Word representations from prediction A second method for generating word representation is inspired from the neural network models used for language modeling. Traditionally neural network language models are given a word and predict context words. This prediction process can be used to learn a representation for each word. When train on a textual corpus, the neural models learn the word representations by starting with a random vector and then iteratively shifting a word's representation to be more like the representation of neighboring words, and less like the representations of words that don't occur nearby.

The development of models of embeddings is an active research area, with new models including GloVe [80] (based on ratios of probabilities from the word-word co-occurrence matrix), word2vec [70] that implements the continuous bag-of-words and skip-gram architectures for computing vector representations of words, or the recently introduced sparse embeddings based on nonnegative matrix factorization [27].

In this work, we choose to use prediction models to infer the word embeddings for two main reasons. First, prediction models are faster to train [52, 70]. Second, these models were shown to yield good results for different prediction tasks, for example it has achieved an accuracy of 97.22% in the task of Part Of Speech (POS) tagging and an F1 score of 89.59% in the task of Named Entity Recognition (NER) [15, 27].

3.2.2 Word2vec

This section describes *word2vec* [70] (W2V), the word embedding model we use in this dissertation. We choose W2V because it has a number of desired properties for the summarization problem we address. First, it is completely unsupervised, which means that we can learn the representation of the words directly from our corpus without the need of a training dataset with manual annotations, which are expensive and require an extensive human effort especially on large datasets. Second, W2V is faster to train than other embedding methods, even on large datasets [71]. Third,

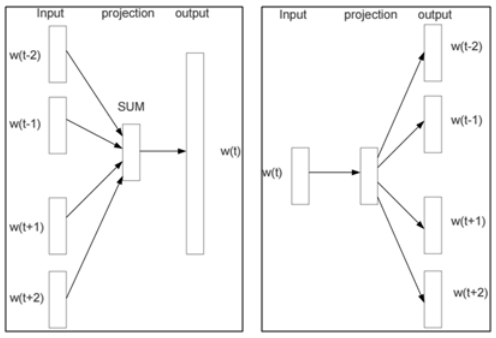


FIGURE 3.5: The CBOW model predicts the current word based on the context (image on the left), and the Skip-gram predicts surrounding words given the current word (image on the right).

the similarity between words is specific to the vocabulary used in movie reviews (as discussed earlier).

W2V was introduced by Mikolov et al. [70]. It is one of the most popular techniques for learning a word embedding over billions of words. When trained on a corpus, W2V learns for each word its representation in a low-dimensional vector space. Words that are used and occur in the same contexts in the original corpus tend to have similar meaning and are mapped to nearby points in this low-dimensional space. We next introduce the two representation models of words used in W2V and explain how we tune W2V parameters for applying it to the IMDB dataset.

3.2.3 Word2vec models

In the original W2V paper [70], the authors introduce two distinct models for representing words in a vector space, i.e., the *continuous bag of words* model (denoted CBOW) and the *skip-gram* model. As shown in Figure 3.5 the skip-gram model predicts the surrounding words given the center words, whereas CBOW predicts the surrounding words given the center word. Algorithmically, CBOW and skip-gram are similar, except that CBOW predicts target words (e.g. “movie") from source context words (“you must go and see this"), while skip-gram does the inverse and predicts context-words from the target words. This inversion has the effect that CBOW smooths over a lot of distributional information (by treating an entire context as one observation). This makes the CBOW useful for smaller datasets and also faster to train than skip-gram. However, skip-gram treats each context-target pair as a new observation, and this tends to perform better with larger datasets. The skip-gram model also represents rare words or phrases well. While CBOW and skip-gram are similar algorithms and produce similar embeddings, it has been shown that skip-gram gives better results than CBOW [70]. Therefore, we decided to use skip-gram in our work.

The Skip-Gram model The skip-gram embedding model seeks to represent each word $w \in V$ and each context $c \in V$ as a d-dimensional vectors $\vec{w}$ and $\vec{c}$, such that words that are “similar” to each other have similar vector representations. It does so by trying to maximize the likelihood of the prediction of contextual words given the center word. More formally, let us consider a corpus of length T and the t-th word w^t, whose index in the vocabulary is j, so we'll call it w_j ($j \in |V|$).

The skip-gram model predicts each neighboring word in a context window of size $2 * L$ words, (L being the size of the window) from the current word. For a context window $L = 2$ the context is $[w^{t-2}, w^{t-1}, w^{t+1}, w^{t+2}]$ and we are predicting each of these words from word w_j. Let's assume that we are predicting one of the $2L$ context words, for example w^{t+1}, whose index in the vocabulary is $k \in |V|$. Computing the probability $p(w_k|w_j)$ is computing the dot product between the vectors of the context vector for w_k and denoted $\vec{c_k}$ and the vector for w_j denoted $\vec{v_j}$. The dot product of $\vec{c_k}$ and $\vec{v_j}$ is not a probability, it is just a number ranging from -∞ to +∞. The softmax function is then used to normalize the dot product into probabilities [70]. Computing this denominator requires computing the dot product between every other word w in the vocabulary with the target word w_i.

$$p(w_k|w_j) = \frac{exp(c_k, v_j)}{\sum_{i \in |V|} exp(c_k, v_i)} \tag{3.4}$$

3.2.4 W2V parameters' setting

W2V relies on a few configuration parameters that we discuss below.

Dynamic window size The *context window* defines the words around a *target* word, which together capture the context of the target word. For example, in the sentence "One of the best **movies** of the year" if we consider **movies** to be the target word and a window of size 2, then the context would be 2 words before the target word i.e. "the best", and two words after the target word, i.e. "of the". The common way of defining a context window is to use constant-sized unweighted window before and after the target word. For instance, if the window size is 5, then any word located five words apart from the target will have the same weight, irrespective of the distance to the target.

Intuitively, words closer to the target are more important. Hence, W2V assigns weights according to the distance between the context word and the target (i.e., words closer to the target have higher weights). When applying W2V to our vocabulary we select a context window size of 10 as in the original W2V paper [70]. First, window sizes larger than 5 lead to higher accuracy [70]. Second, we evaluate the accuracy of the word representation with window sizes of 5, 10, and 15 in our dataset. With a window of size 5, synonyms are mostly typos of a given word and also words of the same "type" only adverbs, for example. With 15, the set of words was more *topical* (related words i.e. car:wheel, instead of synonymous words i.e. car:vehicle), in addition larger windows increase the time to run the algorithm. 10 represents a good compromise for our problem. For example, in Table 3.2 for the word "Character" using a window of 5 we had two typos as similar words i.e. "charactor" and "charactr", with a window of 15 we had more topical words like "likable" and "personality". We use the same weighing as W2V, which weighs words by the distance from the target word divided by the window size. For example, a window of size four will weigh its contexts by $\frac{4}{4}$, $\frac{3}{4}$, $\frac{2}{4}$, $\frac{1}{4}$.

Sub-Sampling In our corpus, the most frequent words occur hundreds of millions of times as we see in Figure 3.6. Such words (stop words for example) usually provide less information than the other words. For example, the co-occurrence of "story" and "the" is less informative for the skip-gram model than the co-occurrence of "story" and "plot". Sub-sampling is a method for damping the effect of very

TABLE 3.2: Extracted 4 most similar words to the word "Character" for different window sizes

5	10	15
charactr	role	characterization
characterization	protagonist	likable
protagonist	characterization	personality
charactor	antagonist	protagonist

frequent words. A byproduct of sub-sampling is to remove stop-words. The sub-sampling method presented in Mikolov et al. [70] randomly removes words that are more frequent than some threshold σ with a probability of p, where $freq_w$ represents the frequency of the word w in the corpus:

$$p_w = 1 - \sqrt{\frac{\sigma}{freq_w}} \tag{3.5}$$

Following the recommendation in Mikolov et al. [70], we use $t = 10^{-5}$. In W2V the removal of words is done before the corpus is processed into (word,context) pairs. This has the effect of enlarging the context window size for many words, because they can now reach words that were not in their original window size. This means that this step adds context-words that are both more informative and far away from the target word, making the similarities more topical. Sub-sampling of the frequent words improves the training speed and makes the word representations significantly more accurate [70].

Rare words' deletion It is a common practice to ignore words that are rare in the training corpus. W2V removes words that are less frequent than a threshold from the corpus before creating context windows. As with sub-sampling, this step reduces the distance between words, inserting new word-context pairs that did not exist in the original corpus with the same window size. In our work, we select this threshold based on the analysis of Figure 3.6. We set the threshold for deleting words to 10, which leaves us with 15% of the vocabulary, the reason behind this choice is that frequent words are more important and too many words make the algorithm slower.

3.2.5 W2V examples

This section illustrates the results of applying W2V to the IMDB review dataset. We train W2V on the vocabulary described in Section 3.2 to generate the word embedding. To visualize the words into the two-dimensional space we use a dimensionality reduction step where word vectors are projected down to 2D vector space. We use the t-distributed Stochastic Neighbor Embedding (t-SNE) [61], which is a dimensionality reduction technique that helps to visualize large real-world datasets with limited computational demands.

We have arbitrarily selected four seed words "Story, Special-effect, Actor, Score" and for each seed word we have extracted the set of similar words. In Figure 3.7, we show a 2D projection of this set of words. We observe that words that are semantically similar are close to each other and they end up being clustered nearby each other (we used distinct colors to differentiate each cluster). For example, the

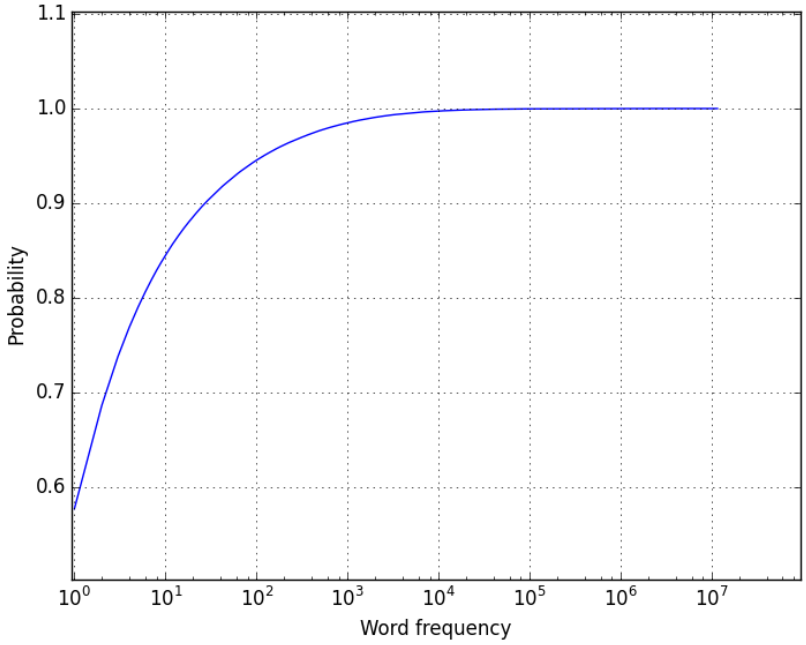


FIGURE 3.6: Word frequency in IMDB dataset

cluster in red is about “Special-effect", whereas the cluster in blue is about “Story". We can also see that some words are very close to each other, for example the words “supporting-actor”, “supporting-cast” and “supporting-player” almost all overlap due their semantic similarity.

3.3 Semantic similarity

The result of applying W2V as discussed in the previous section, is word embedding, where we represent each word as a vector. This section explains how we use this embedding to compute the similarity between words.

Our problem requires similarity between words in different use cases, which calls for different types of “similarity”:

- **Identification of similar words**: Given a seed word that represents a movie aspect (for example, “Cast”), the goal is to identify other words in the corpus that have a similar meaning or are used to describe similar aspects. We call the set of words that describe an aspect the *aspect signature*.
- **Similarity between a sentence and a set of similar words**: Given the aspect signature, we must identify sentences that talk about that aspect. In this case, we must compute the similarity between each sentence and each aspect signature to identify the set of aspects discussed in each sentence.

We explain in details how we compute each similarity in the following subsections.

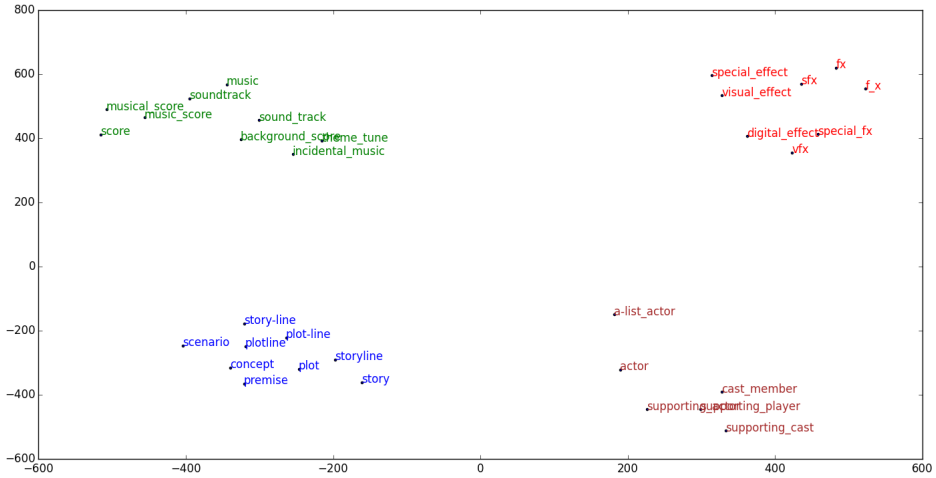


FIGURE 3.7: Clusters of semantically similar words emerge when the word2vec vectors are projected down to 2D using t-SNE.

3.3.1 Word Vector Similarity

To define the similarity between two words w_1 and w_2, we need a metric of vector similarity. By far the most common vector similarity metric is the cosine of the angle between the vectors [64]. The cosine vector similarity metric is based on the dot product. The dot product acts as a similarity metric because it tends to be high when the two vectors have large values in the same dimensions. The dot product is biased by the vector length as it gets higher if a vector is longer. The simplest way to solve this problem is to normalize the dot product by the lengths of each of the two vectors, which is the cosine measure. To compute the similarity of two word vectors $\vec{v_1}$ and $\vec{v_2}$ we use the following formula:

$$cosine(\vec{v_1}, \vec{v_2}) = \frac{\vec{v_1}.\vec{v_2}}{|\vec{v_1}||\vec{v_2}|} = \frac{\sum_i^n v_{1i} v_{2i}}{\sqrt{\sum_i^n v_{1i}^2}\sqrt{\sum_i^n v_{2i}^2}} \tag{3.6}$$

The cosine value ranges from 1 for vectors pointing in the same direction to -1 for vectors pointing in opposite directions, passing through 0 for vectors that are orthogonal. Given the W2V embedding, we illustrate the use of the cosine similarity to list the most similar words to three example words from our corpus: “good”, “funny”, and “kid”. Table 3.3 shows the ten most similar words to each of these seed words. The scores represent the cosine similarity between the seed word and the candidate word. These examples show that the similarity scores capture the semantic relationship between two words in some cases, for example, “great" and “good" or “amusing" and “funny" have cosine similarity above 0.7 as we would expect as they are synonyms. We also see, however, that similarity is high for antonyms. The W2V embedding will cause words that appear in similar contexts to be close, which causes synonyms as well as antonyms to have high cosine similarities. For example, “bad" and “good" have 0.63 similarity. This problem only happens with adjectives.

TABLE 3.3: Examples of the closest words to a given word using the Word2Vec model.

good	picture	kid
decent 0.81	film 0.74	child 0.75
great 0.76	movie 0.66	teenager 0.75
nice 0.68	motion-picture 0.6	youngster 0.71
solid 0.68	pic 0.58	boy 0.68
bad 0.67	flick 0.54	toddler 0.63
fine 0.67	production 0.47	lad 0.63
top-notch 0.65	picture-nominee 0.47	teenage-boy 0.63
terrific 0.64	opus 0.46	kiddy 0.63
fantastic 0.62	animated-feature 0.46	teenage-girl 0.6
well-done 0.61	historical-epic 0.46	tyke 0.6

3.3.2 Similarity Metrics on Vector Sets

In addition to computing the similarity between two words represented as vectors, we must also compute the similarity between an aspect signature and a sentence (i.e., two sets of vectors). Take for example vectors X and s. Where $X = \{w_1, ..., w_t\}$ and $s = \left\{w'_1, ..., w'_f\right\}$, t and f being respectively the sizes of X and s. This section discusses similarity metrics for sets of vectors from the literature.

Matching coefficient The first similarity metric, the matching coefficient [64], simply counts the number of matching words in both sets of words without considering the length of the sets. For example lets assume that we want to measure the similarity between a set $X = \{\text{“}movie\text{”}, \text{“}flick\text{”}\}$ and a sentence s= “The film was boring but the special-effects were mind-blowing". In this example our goal is to match the set of nouns in s, say $Y = \{\text{“}film\text{”}, \text{“}special - effects\text{”}\}$, with at least one term from X, which we can compute with the intersection, $X \cap Y$. The matching coefficient would be of 0 as there is no matching term between X and Y.

Jaccard coefficient The Jaccard coefficient [64], ranges from 0 (no overlap) to 1 (perfect overlap). It's normalized by the total number of words in both sets. We compute the Jaccard coefficient as follows:

$$\frac{X \cap Y}{X \cup Y} \tag{3.7}$$

In our example, the Jaccard coefficient would be equal to 0 again as there is no matching term between X and Y.

max-pairs We define the similarity between X and Y as the maximum similarity between each vector pair $(\vec{x_i}, \vec{y_j})$ of X and Y:

$$\underset{max-pairs}{similarity}(X, Y) = \max_{i,j}(cosine(\vec{x_i}, \vec{y_j})) \tag{3.8}$$

The main advantage of the max-pairs distance metric is that it leverages the word embedding in the vector space. For example, even tough “film” is similar to “movie” it was not taken into consideration in neither the matching nor the Jaccard coefficients because it is not an exact match. Additionally, it computes distances between

sets of vectors without ever learning a new representation for each set of vectors X and Y.

max-count-pairs Given two set of vectors X and Y their similarity is defined as the normalized sum of the maximal similarities of all vectors $\vec{x_i} \in X$ with the vectors $\vec{y_j} \in Y$.

$$\underset{max-count-pairs}{similarity}(X, Y) = \frac{1}{n} \sum_{i}^{n} \max_{j}^{m}(cosine(\vec{x_i}, \vec{y_j})) \tag{3.9}$$

3.4 Summary

In this chapter, we introduce the IMDB dataset that we use in the rest of this dissertation. We show that the number of reviews and ratings is skewed, which implies that we must evaluate our methods for different popularity groups of movies. We also explain the three steps—namely, tokenization, lemmatization, and collocation detection—we follow to clean the noisy reviews in the movie dataset in order to define the final vocabulary V that we use throughout this dissertation. We present different approaches to learn word embeddings, and explain the model we use to represent words as vectors: W2V. W2V is completely unsupervised, fast to train even on large datasets, and able to learn word embedding based on the words' context. More specifically, we use W2V's skip-gram model to learn the words' embedding. Finally, this chapter introduces the three similarity metrics we use in the dissertation. In the next chapter, we rely on the cosine similarity to compute the similarity between words to build the aspect signatures. In Chapter 5, we evaluate both "Max pairs" and "Max count pairs" to compute the similarity between a sentence and the aspect signature.

Chapter 4

Movie Aspect Signature Detection

The objective of this dissertation is to create personalized summaries that contain information about the *aspects* a user is interest in. We use the term *aspect* to describe the characteristics/features of a movie that users can comment on, i.e. "Story", "Visual effects", or "Cast". In this dissertation, aspects are defined as the union of the Oscar categories[1] and the IMDB movie genres. The full list of aspects is available in Table 4.1. We believe that the Oscar categories lead to meaningful movie characterization because they are selected by domain experts. These categories are also well-known by the public given the popularity of the Academy Awards. IMDB genres complement these aspects with genres that most people will use to express their movie preferences. We consider these aspects (i.e. Oscar categories and the IMDB genres) as *seed terms* for our aspect signatures detection algorithm.

Figure 4.1 illustrates the difficulty of annotating review sentences with aspects in an automated way. It shows a review covering different movie-related aspects such as "Sound-effects" , "Visual-effect", and "Cast". We observe that within a review users use different nouns to refer to the same aspect. For example, `CGI` (computed generated imagery) and `Special-effects` are used for "Visual-effect". Moreover, the same noun is used to refer to multiple aspects. For example, `CGI` refers both to "Animation" and "Visual-effects".

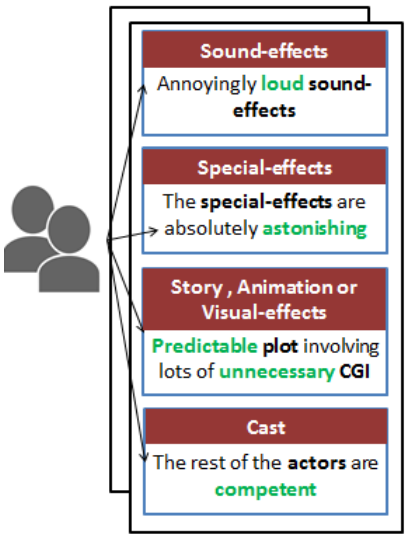
Unlike structured reviews in which users comment and/or rate products or services for pre-defined aspects (as we can find in reviewing websites, i.e., Beeradvocate[2] and Tripadvisor),[3] movie reviews as found on IMDB and Rotten Tomatoes are unstructured. A naive approach to identify aspects is to search for sentences that explicitly use the aspect name. Looking for the exact match of a single word is too limiting as reviewers use a large vocabulary to describe each aspect. For example, words actor and performer are both used to describe the aspect "Cast".

Our problem is to develop an algorithm to identify the set of words that are specific to each aspect in reviewers' vocabulary. We call this set of words the *aspect signature*. It is defined for a review corpus. The aspect signature includes noun words. Recent research shows that 60-70% of the aspect signature are explicit nouns [38].

[1] The Academy of Motion Picture Arts and Sciences delivers the Oscar Awards in 11 categories.
[2] https://www.beeradvocate.com/
[3] https://www.tripadvisor.com/

TABLE 4.1: List of the aspect seed terms

Aspect seed term list
visual-effects, story, sound-effects, script, score, makeup, cinematography, cast, animation, comedy, action, thriller, western, news, director, science-fiction documentary, romance, horror, adventure, fantasy, biography, war, crime, musical



Aspects	Aspect signature	
	Terms	Opinion-words
Sound-effects	-Audio-effects -Foley	-Loud -Stunning
Visual-effects	-Special-effects -CGI	-Mind-bending -Unnecessary
Story	-Story-line -Plot	-Predictable -Long
Cast	-Actor -Performer	-Competent -Charismatic
Animation	-CGI -Puppetry	-Astonishing -Unnecessary

FIGURE 4.1: Example of aspect signatures extracted from review sentences

For this reason, we decide to focus on noun extraction to build aspect signatures. We use adjectives to understand the opinions that describe these aspects.

In the remaining sections of this chapter we define the problem of aspect signature detection, and propose an aspect signature extraction algorithm. Finally we conduct a thorough experimental evaluation.

4.1 Tradeoffs in signature definition

Our goal is to define a signature per aspect, made of all nouns used to describe this aspect in users' reviews. Before designing our algorithm, we must first decide: (1) If all signatures have the same length (number of words)(2) If some words can belong to multiple aspect signatures.

4.1.1 Aspect signature length

The higher the number of words in the signature the easier it will be to find sentences, if we have the same number of words in signatures of all aspects then annotating would be more uniform across all aspects. Therefore, the length of the aspect signature k determines how likely we will annotate a sentence with an aspect. Some aspects are more popular than others and the vocabulary used to describe them is richer (e.g. both aspects "Adventure" and "Action"). We conduct a simple experience to assess the impact of a fixed signature length across aspects. We extract the semantically top k closest terms to each aspect seed term using the cosine-similarity (see definition 3.6), and we vary the value of k across aspects. For each top k terms per aspect we compute the similarity value of the closest term to the seed term (i.e. *most similar*) and the most distant term (i.e. *least similar*). Table 4.2 shows that the similarity of the least similar term significantly varies for different Oscar aspects. For example, for a fixed length *k=10*, the least similar term to the aspect "Visual-effects" has a similarity of **0.35** whereas "Cinematography" **0.73**. Also for a fixed length of *k=40*, the least similar term to the aspect "Visual-effects" has a similarity

TABLE 4.2: Max/Min similarity of noun terms to the seeds of Oscar aspects for different signatures' length

	k=10		k=20		k=30		k=40	
	Most similar	Least similar	Most similar	Least similar	Most similar	Least similar	Most similar	Least similar
Visual-effects	0.5	0.35	0.5	0.34	0.5	0.31	0.5	0.31
Story	0.82	0.55	0.82	0.5	0.82	0.47	0.82	0.46
Sound-effects	0.63	0.53	0.63	0.5	0.63	0.48	0.63	0.47
Script	0.89	0.57	0.89	0.5	0.89	0.48	0.89	0.45
Score	0.88	0.61	0.88	0.56	0.88	0.55	0.88	0.53
Makeup	0.96	0.55	0.96	0.51	0.96	0.48	0.96	0.46
Cinematography	0.94	0.73	0.94	0.65	0.94	0.61	0.94	0.59
Animation	0.77	0.62	0.77	0.55	0.77	0.52	0.77	0.51
Cast	0.84	0.52	0.84	0.45	0.84	0.42	0.84	0.4

of **0.31** whereas for "Cinematography" it's **0.59**. Using a fixed length k for all aspects would lead to signatures with terms that may be semantically different from the seed term of the aspect.

An alternative to fixed length signature is to consider a similarity distance from the seed term. Typically, the signature length may vary depending on the density of terms around the seed term.

An absolute similarity threshold, however, may not work across aspects. Table 4.3 studies different similarity thresholds. A similarity threshold of 0.45 (which is a high similarity threshold) from the seed term gives 110 terms in the signature of "Thriller" but five terms for "Visual-effects". Hence, choosing an absolute similarity threshold does not guarantee that each aspect signature has enough terms.

Therefore instead of computing a similarity distance term, we compute a *similarity distance*, denoted as d, from the most similar term to the seed up to the least similar term included in an aspect signature. In this way, regardless of the density around a seed term we are able to extract variable length signatures with semantically equivalent terms for all aspects. We have shown the difficulty of choosing a fixed length signature. In the next section, we show how signatures overlap impact that decision.

4.1.2 Overlap between signatures of different aspects

We study the aspect signatures overlap while varying the similarity distance d from the seed term of each aspect. Figure 4.2 shows the number of aspect signatures that overlap (y axis) for four different cases: (a) aspect signatures don't overlap with each other (blue line), (b) the number of overlapping terms between signatures is one term (green line), (c) the number of overlapping terms is two terms (red line) and (d) the number of overlapping terms is four terms or more (black line).

Intuitively, we expect the terms that have a higher similarity distance from the seed terms to overlap with other aspect signatures. For d=0.1, we have 21 non-overlapping signatures of aspects (out of 25) and 4 aspects that have only one term in common with other aspects. The overlapping term at d=0.1 between the aspects "Sound-effects" and "Score" is "Soundtrack". When d increases, the number of overlapping aspect signatures grows as well as the number of terms that overlap between signatures. For example at d=0.25 all aspect signatures overlap, with 15 aspect signatures that have more than 3 terms that overlap.

TABLE 4.3: Length of noun aspect signatures per similarity threshold

	0.1	**0.15**	**0.2**	**0.25**	**0.3**	**0.35**	**0.4**	**0.45**
Visual-effects	0	0	1	2	3	3	4	5
Story	0	0	1	4	7	10	14	24
Sound-effects	0	0	0	0	0	0	2	8
Script	0	1	2	3	4	9	15	27
Score	0	2	4	7	12	20	30	42
Makeup	0	0	0	1	2	3	5	11
Cinematography	1	2	5	10	18	34	61	103
Cast	0	0	1	2	4	8	14	20
Animation	0	0	0	1	2	5	14	25
Comedy	0	0	0	1	6	12	22	42
Action	0	0	0	0	0	0	1	4
Thriller	0	1	4	9	20	42	68	110
Western	0	0	0	0	0	0	1	4
News	0	0	0	0	0	0	0	2
Documentary	0	0	0	0	0	2	5	12
Romance	0	0	0	0	0	0	1	3
Horror	0	0	0	1	2	4	8	20
Adventure	0	0	0	0	0	0	0	2
Fantasy	0	0	0	0	0	0	2	8
Biography	0	0	0	0	1	5	10	15
War	0	0	0	0	2	4	9	20
Crime	0	0	0	0	0	2	6	13
Musical	0	0	0	0	0	0	1	4

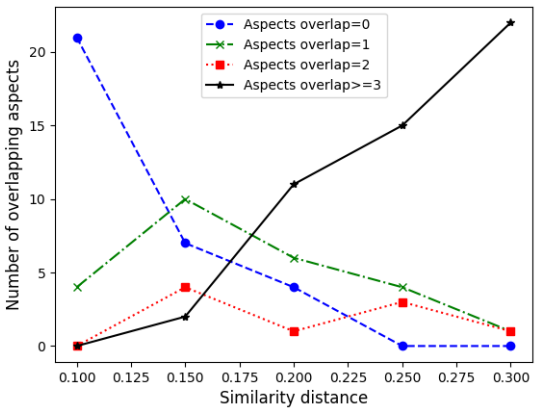
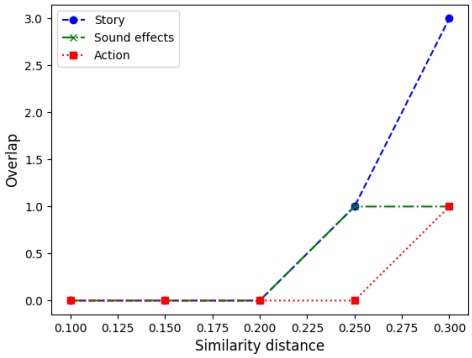


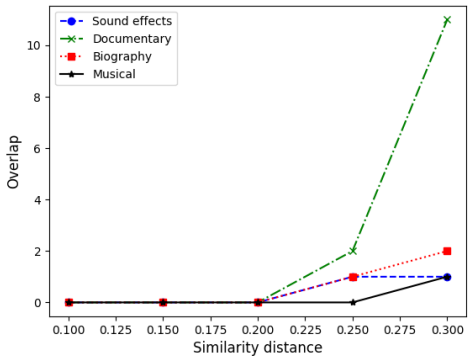
FIGURE 4.2: Number of overlapping aspect signatures per similarity distance of nouns from seed terms

Although we want to minimize overlaps between signatures, the overlap is unavoidable. There exists few aspects that keep overlapping with each other at a similarity distance $d >= 0.25$ such as "News" (see Figure 4.4a) while most of the aspects tend to overlap with multiple other aspects even at a similarity distance $d < 0.25$ i.e. "Action" or "Adventure", Figures 4.5a and 4.5b. A small similarity distance d minimizes the number of overlapping aspect signatures, but it also reduces the number of terms included in the aspect signature. Very small signatures of aspects would result in having few or no sentences to pick from to generate our summaries. On the other hand, a high similarity distance d increases the number of terms in aspect signatures and leads to a higher overlap between aspects. This shows that controlling overlap between aspect signatures is hard. The challenge is to find a good compromise between having enough words in the aspect signature and making sure that aspect signatures are distinctive. We study this issue in Chapter 5 and propose solutions.

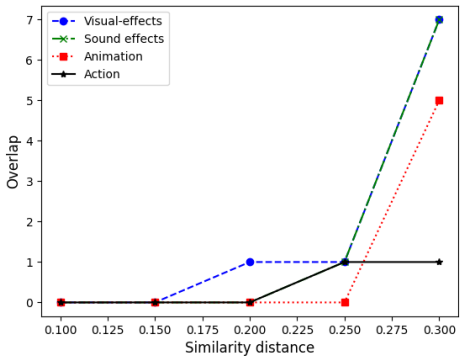


(A) Script.

FIGURE 4.3: Distribution of overlapping nouns in pairs of aspects

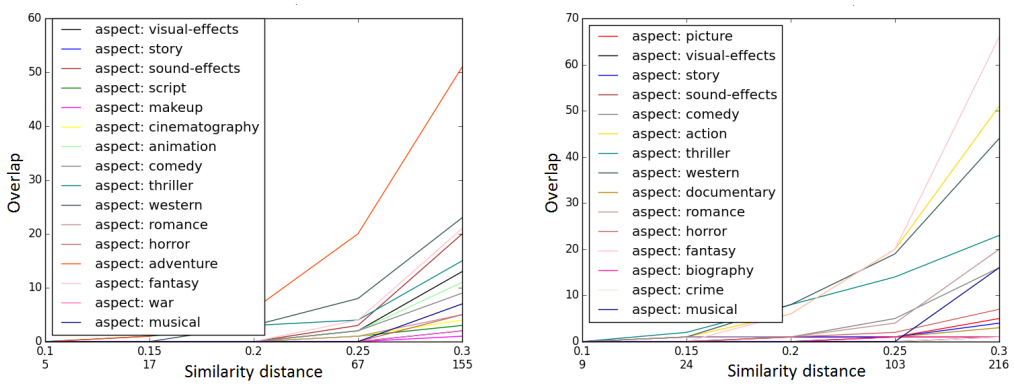


(A) News.



(B) Makeup.

FIGURE 4.4: Distribution of overlapping nouns in triads of aspects



(A) Action. (B) Adventure.

FIGURE 4.5: Distribution overlapping nouns in higher arity sets of aspects

4.2 Aspect signature detection

4.2.1 Problem definition

In this section, we formally define the *aspect signature detection* problem. From a unique seed term that is representative of an aspect , we must identify in the corpus of reviews all terms that represent the same aspect. We use the W2V embedding and the cosine similarity defined in Chapter 3 to find the words that are semantically related to the seed term for each aspect.

As discussed in the previous section, it is important to carefully control both the *similarity distance* (between words in the same signature) and the *degree of overlap* (across signatures) of aspect signatures. These requirements are not addressed in related methods in the literature [38, 106].

Problem Statement 4.2.1. We denote as $A = \{a_1, \ldots, a_n\}$ a set of aspects identified by their corresponding seed terms a_i, and as $ASN(a_i)$ the list of noun terms in the signature of an aspect a_i. The similarity of two terms is defined as the cosine similarity of the corresponding vectors in the W2V representation of a review vocabulary V (see Equation 3.6).

Given a set of seed terms per aspect $A = \{a_1, \ldots, a_n\}$, and a vocabulary V of terms represented as vectors $\vec{t} \in \mathbb{R}^{dim}$ extracted from a review corpus, the *signature of an aspect* $a_i \in A$, denoted as $ASN(a_i)$, is composed of a list of noun terms $ASN(a_i) =< n_{i1}, ..., n_{ik} >$ such that $\forall n_{ij} \in ASN(a_i)\ \exists \vec{t_{ij}} \in \mathbb{R}^{dim}$.

- the signatures have a *fixed similarity distance*: $sim(ASN(a_i)[1], ASN(a_i)[k]) < d)$ where k is the least similar term in the noun or adjective signature of an aspect a_i;
- the terms have a *fixed signature overlap*: $\forall \vec{t} \in \bigcup_{a_i \in A} ASN(a_i) \sum_{j=1}^{n} (\{\vec{t}\} \cap ASN(a_i)) <= \alpha$ where n is the total number of aspects.

As explained earlier, these two constraints control the quality of the aspect signatures. In a nutshell, we are interested in employing *distinctive enough* and *semantically related* lists of terms per aspect in order to discover the aspects in unstructured reviews in a generic way that covers different writing styles (with simple or complex sentences) and movies. We select d and α parameters empirically as we show in Chapter 5.

Algorithm 1: Aspect signature extraction algorithm

Input : Word embedding matrix $W \in \mathbb{R}^{dim*h}$
List of aspect seed terms A= $\{a_1, a_2, ..., a_n\}$
α the degree of overlap
d the similarity distance

Output: The Aspect Signature Matrix ASM, where each row i contains the signature of nouns of the aspect a_i $ASN(a_i)$

mostSimilarTerms ← getMostSimilarTerms(W,A, d) /* This method returns a hash table where each key is an aspect and each value a sorted list of <$\vec{t_j}$, sim($\vec{t_j}$, $\vec{a_i}$)> pairs s.t. $\forall$ $\vec{t_j}$, sim($\vec{t_j}$, $\vec{a_i}$)<d */

ASM= init() /* Initialize the Aspect Signature Matrix with zeros */

for a_i *in* A **do**
  for <$\vec{t_j}, sim(\vec{t_j}, \vec{a_i})$> *in* $mostSimilarTerms[a_i]$ **do**
    ASM[i][j] ← $sim(\vec{t_j}, \vec{a_i})$

for $j = 0$ *to* m **do**
  /* m is the number of columns of the matrix ASM */
  if *countNonZero(ASM[][j])* > α **then**
    /* countNonZero(l) is a function that returns the number of non zero elements for a given list l */
    sortedIndexes ← sortDscIndexes(ASM[][j])
    for *aspectIndex in sortedIndexes[:n-*α*]* **do**
      ASM[aspectIndex][j] ← 0
  return ASM

	Storyline	Plot	Screenplay
Story	0.67	0.8	0
Script	0	0.7	0.65

FIGURE 4.6: Example of the second step of the algorithm

	Storyline	Plot	Screenplay
Story	0.67	0.8	0
Script	0	**0**	0.65

FIGURE 4.7: Example of the third step of the algorithm

4.2.2 Algorithm design

In this section we describe our algorithm to extract aspect signatures. Algorithm 1 outlines the core steps involved in the aspect signature extraction. The inputs to the algorithm are: the W2V representation of the vocabulary of our review corpus W, the list of aspects A, the similarity distance d and the degree of overlap α . The output of the algorithm is the Aspect Signature Matrix ASM, where each row i contains the signature of nouns of the aspect a_i, $ASN(a_i)$. Each term in $ASN(a_i)$ has to verify two constraints, first it should be at a maximal distance d from the seed term a_i, second it should belong at max to α aspect signatures.

The algorithm starts by initializing the ASM with zeros. The function getMostSimilarTerms(W,A,d) extracts, for each aspect seed term a_i, terms that are exactly at a similarity distance d. The loop in lines 3 to 5 fill the cells of the ASM. The values in the cells correspond to the cosine similarity between the vector of the i^{th} aspect and the vector of j^{th} most similar term.

In lines 7 to 10 of the algorithm, we check the second constraint α. We loop over the columns of the ASM and we check if the size of the non zero elements in each column $countnonzero(ASM[][j])$ exceeds α. If it is the case, this means that the j^{th} term belongs to more than α aspect signatures and that the second constraint of the degree of overlap is violated. To solve this issue, we use the function $sortDscIndexes(ASM[][j])$ that returns the indexes that sort the column $ASM[][j]$ in a descending order. $sortedIndexes[: n - \alpha]$ returns the list of aspect indexes with the lowest similarity values to the j^{th} term. We replace the cells of each aspect index $aspectIndex$ in this list with zero (lines 9 to 10). The algorithm returns the aspect signature ASM in line 10.

We use the following example to illustrate how algorithm 1 works. For this purpose we take d equals to 0.15 and α equals to 1. We consider two aspect seed terms "Story"and "Script" i.e. A={ "Story", "Script"} . Figure 4.6 represents the ASM after filling it with the most similar terms at a distance d for both aspects (lines 3 to 5). We observe that for the term "Plot" in the third column, the number of non zero elements is equal to 2 that exceeds $\alpha = 1$. This is an issue because it means that the term "Plot" belongs to 2 aspect signatures i.e. "Script" and "Story". $sortDscIndexes(ASM[][1])$ returns the indices that sorts $ASM[][1]$ in a descending order: [0,1], $n-\alpha$=1 (n=2 which is number of aspects/rows) so $sortedIndexes[: n-\alpha]$ = 1 which represents the index of the aspect having the lowest similarity with the term "Plot" i.e. "Script". We replace the value of the cell $ASM[1][0]$ with zero as one can see in red in Figure 4.7. The algorithm returns the ASM (Figure 4.7), because both of our constraints are verified.

TABLE 4.4: Length of extracted noun signatures

	0.1	0.15	0.2	0.25	0.3
Visual-effects	5	6	12	24	38
Story	3	3	3	5	10
Sound-effects	8	17	49	105	242
Script	1	1	1	5	7
Score	3	4	5	7	9
Makeup	1	1	3	8	20
Cinematography	1	3	5	8	15
Cast	1	2	4	6	6
Animation	3	7	10	21	46
Comedy	6	10	17	31	58
Action	5	17	29	67	155
Thriller	5	11	21	24	37
Western	5	19	35	87	209
News	8	16	30	72	186
Documentary	2	6	21	32	60
Romance	5	11	27	62	149
Horror	1	4	8	14	23
Adventure	9	24	51	103	216
Fantasy	6	16	39	88	223
Biography	3	4	6	19	32
War	3	5	12	36	97
Crime	4	7	21	42	83
Musical	7	18	44	106	236

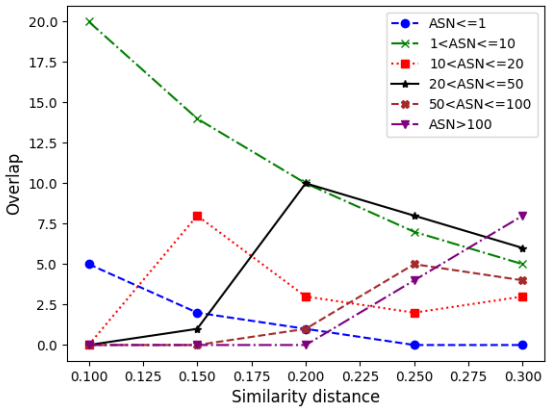
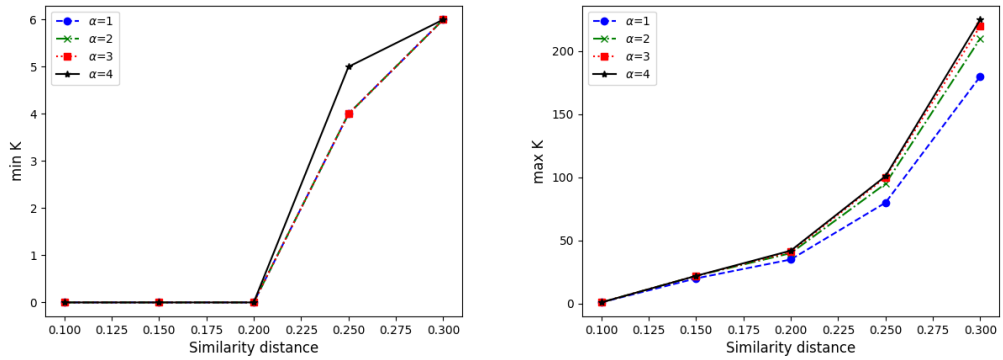


FIGURE 4.8: Length of aspect signatures per similarity distance

Parameter setting

The aspect detection algorithm involves two parameters to be set empirically namely α and d. In this section we explain how we set the ranges for α and d. The algorithm will be fine tuned in Chapter 5. As we can see in the frequency distributions of the overlapping signatures (Figure 4.2) and their corresponding length (Figure 4.8) the interesting range of similarity distance values d for all aspects lies between 0.1 and 0.2. The reason is that we observe that even thought the overlap between signatures is minimized when d equals to 0.1 (4 out of 25 signature overlap), we obtain signatures that do not contain enough terms (i.e. 12 out of 25 extracted signatures have length less than 3). On the other hand, for d above 0.2, signatures' overlap a lot (i.e. 17 out of 25 aspects share common nouns which means that almost all aspects would overlap with each other, this would make the aspect signatures not distinctive enough for each aspect). Table 4.4 represents the length of extracted signatures for different similarity distances d, and we observe that for $d > 0.2$ we add too much noise to the noun aspect signatures, for example the length of the aspect signature Sound-effects distance $d = 0.25$ is equal to 105. We also observe that choosing a value below 0.15 for example 0.1 would lead to having a very low number of aspect noun signature per aspect. Given that the signature extraction algorithm presented in Section 4.2.2 is able to control the overlap among signatures, we experiment with only two values of similarity distance namely $d = 0.15$ and $d = 0.2$.



(A) Minimum Length. (B) Maximum length.

FIGURE 4.9: Length of extracted noun signatures when varying α

In order to decide on the degree of overlap, α, we can tolerate among the noun signatures of different aspects, we run our algorithm with values of d ranging from 0.1 to 0.3 and α varying from 1 to 4. Then, we measure the maximum and minimum length, k, of the extracted signatures. As we can observe in Figures 4.9, when $\alpha > 2$ there is no impact on the minimal nor the maximal length for noun signatures. For this reason, we decide to experiment only with two α values, namely 1 (i.e., no-overlap) and 2 (i.e., only overlapping pairs).

Qualitative analysis

After selecting ranges of parameter values in the last section, we now analyze the aspect noun signatures generated by the algorithm.

Table A.1 in Appendix A presents the aspect signatures (with $\alpha = 1$) of all aspects generated by our algorithm for a similarity distance of d=0.15. Despite the disjointness constraint that might minimize the number of terms in signatures, our

TABLE 4.5: Statistics of generated aspect noun signatures for $\alpha = 1$, similarity distance $d = 0.15$

	Length	Max terms frequency	Min terms frequency	Median terms frequency
Visual-effects	6	67346	15	365
Story	3	205615	28409	34981
Sound-effects	14	763	22	2139649
Script	1	39649	39649	39649
Score	4	26308	986	5221
Makeup	1	477	477	477
Cinematography	3	13377	782	4934
Cast	2	16199	1056	8627.5
Animation	6	30584	28	903
Comedy	10	53919	282	2727.5
Action	16	36402	16	450
Thriller	11	46491	61	557
Western	15	61005	42	265
News	16	5098	139	565
Documentary	5	500	65	312
Romance	11	65811	55	683
Horror	4	9815	40	1615
Adventure	19	25471	16	1427
Fantasy	15	7668	35	545
Biography	4	2837	573	1230.5
War	5	2430	21	1404
Crime	7	39983	21	356
Musical	17	88684	53	1147

algorithm is able to extract terms that are relevant to each aspect. We examine the frequency of the terms included in the aspects signature in the review corpus. Table 4.5 shows the minimal, maximal and median number of signatures length using as an example =1 and d=0.15. We observe that our algorithm successfully extracts a good variety of terms, including infrequent ones. Rather than typos, these infrequent terms are usually compound terms or abbreviations. For example, for "Horror" gothic-horror, for “Performance” is perf. Unlike other aspect signature extraction methods, which are based on the popularity of the terms [48, 106], our method is able to mine semantically related terms of varying frequency in our review corpus. This is crucial in order to later build personalized summarization on different aspects that are not necessarily popular in the review corpus and thus described with less frequent terms. We should finally recall that idiomatic terms (e.g., vfx or fx for "Visual-effects") are not found in general purpose dictionaries and thesauri. Also when we analyze the maximal terms frequency per aspect we can see that this frequency varies according to the popularity of the aspects in the corpus, for example the aspect “Documentary” the frequency of the maximal term is 500 in the whole corpus which is a low frequency but this is due to the fact that we only have 15 movies that belong to the category “Documentary”. This is not the case for the aspect “Story” where the frequency of the most popular term in the signature is 205615, as this aspect is discussed in many sentences in our corpus.

4.3 Conclusion

In this chapter, we propose our algorithm to extract aspect signatures from the corpus of reviews. Our analysis shows that using fixed length signatures as a parameter to the algorithm leads to some signatures with terms that are semantically different. The analysis also shows that using an absolute similarity distance from the seed terms does not guarantee that the signatures contain at least one term per aspect signature. We use a similarity distance, d that controls that signatures have enough terms, and are semantically related. We define α, a parameter that controls the degree of overlap between terms and aspect signatures. We run our algorithm with different combinations of values of parameters d and α. Our quantitative analysis shows that the interesting range of similarity distance values, d, for all aspects is between 0.15 and 0.2. Using this range minimizes the overlap between aspect signatures and ensures that each signature contains enough terms. We show that there is a no impact on the minimal nor maximal length of aspect signatures for α above 2. Therefore we decide to experiment with only two α values: 1 (i.e., no-overlap) and 2. In the next chapter, we evaluate the signatures generated with all combinations of $d = 0.15$ and 0.2 and $\alpha = 1$ and 2 for the task of annotating sentences with aspects.

Chapter 5

Annotation of Sentences with Aspects

In this chapter, we rely on the noun signatures extracted in the previous chapter to identify the aspects discussed in each sentence of the reviews. This task of annotating sentences with aspects is challenging. First, some sentences contain more than one aspect [37]. As we can see in Figure 4.1 the third sentence "Predictable plot involving lot of CGI" covers both aspects "Story" and "Visual-effects". Second, the set of terms extracted in the aspect signatures is different from those contained in sentences from reviews. Even though we extract signatures from reviews, we find the set of terms in the signature of each aspect by looking for terms that are similar to the seed. The vocabulary in reviews is rich, so reviewers may use other terms that are similar to terms in the signature but not as similar to the seed. Instead of relying on the exact match between terms in the aspect signatures and the terms in sentences, we conceive similarity metrics between the review sentences and the aspect signatures to annotate sentences with aspects. This more flexible annotation method permits covering different types of movies and review writing styles. Section 5.1 formally defines the problem of *sentence annotation* and the similarity metrics we evaluate.

The evaluation of whether our method annotates sentences with the correct set of aspects requires a reference dataset, where humans manually label sentences with aspects. Section 5.2 presents the user study we conduct to obtain this reference dataset, where we select a random sample of sentences from our corpus and ask human workers to manually annotate these sentences. Sentence annotation is sometimes subjective and workers may disagree on the correct annotation. Section 5.3 analyzes the resulting dataset to characterize the level of agreement among workers and Section 5.4 defines the metrics to evaluate our sentence annotation method. We then rely on the reference dataset with manually labeled sentences in Section 5.5 to tune the parameters of the aspect extraction algorithm, and to evaluate the similarity metrics for annotating sentences with aspects. After carefully tuning our methods using the reference dataset, we compare the accuracy of our sentence annotation algorithm with that of existing methods in Section 5.6 and then we apply the methods to the entire corpus of reviews in Section 5.7. This large-scale analysis sheds light on the number of sentences annotated with each aspect for each movie. The more sentences per movie and aspect, the richer the input for our summary generation algorithm discussed in the next chapter. In addition to identifying the aspects discussed in each sentence, we must identify which opinion the reviewer expressed about each aspect to summarize the main opinions about aspects. Section 5.8 explains the method to select the opinions expressed about aspects.

5.1 Problem Definition

In this section, we define the notation and formalize the problem of sentence annotation.

Problem Statement 5.1.1. $\forall m \in M, \forall r \in R_m, \forall s_j \in S_r$ the goal is to identify $A(s_j)$ which is the set of aspects discussed in sentence s_j s.t. $A(s_j) \subset A$. Given the noun signature $ASN(a_i) =< n_{i1}, \ldots, n_{ik} >$ of an aspect a_i and the noun terms $SN(s_j) = \{n_{j1}, \ldots, n_{jm}\}$ of a sentence s_j, we consider that the aspect a_i is discussed in the sentence s_j, which we denote as $a_i \in A(s_j)$, if $similarity(ASN(a_i), SN(s_j)) >= \theta$.

We rely on the similarity metrics between a set of vectors from Chapter 3 to assess the similarity of noun terms between sentences and aspects. The "Max-pairs" similarity from Equation 3.8 considers that the distance of a sentence to an aspect signature is given by the noun term of the sentence that is the most similar to one of the terms of the noun signature. "Max-count-pairs" similarity (Equation 3.9) relies on the assumption that several nouns of a sentence (especially for large sentences) may be close to at least one noun of an aspect signature. It essentially normalizes the sum of the similarity values of closest pairs of distinct noun terms with the total number of nouns of a sentence. As we will see in Section 5.4 these similarity metrics are more effective compared to existing methods such as the matching coefficient from Equation 3.7, which is frequently used in sentence annotation [106], or a simple count of matching terms [98].

5.2 Crowd-Sourced User Study

Evaluating any method to annotate sentences with aspects is challenging due to the lack of ground truth. To overcome this challenge, we conduct a user study where users manually annotate random sentences from our corpus. We sample 600 sentences across our review corpus according to their length. We analyze the distribution of sentence length in Figure 3.4b, and split the sentences into three ranges according to percentiles: we pick 200 sentences from the ranges 1-15 words; 200 from the ranges 15-24 words; and the remaining 200 sentences have a number of words greater than 24. The resulting set of sentences comes from reviews of 563 different movies.

We recruit workers on the Prolific Academic platform to manually annotate each sentence with one or more aspects related to Oscar categories and movie genres. This platform allows the selection of workers according to some criteria. The only criteria we chose is English fluency to make sure that all workers correctly understand the task and the provided sentences. We obtain 180 workers to annotate the 600 randomly selected sentences. We ensure that three different workers annotate each sentence.

We follow all known best practices in designing crowd-sourced tasks in order to minimize random answers that would bias our results. Before starting the task, we asked workers to pass an entry test where they had to annotate few simple sentences that we have annotated in advance. If workers were unable to give the correct annotations they failed in the test and were denied access to the remaining annotation task. Only three participants failed the entry test.

Figure 5.1 shows a screen shot of the web page for the user study where we present the instructions to follow in the beginning of the page, then we present the aspects' definitions (when participants hovered the mouse pointer over the name of

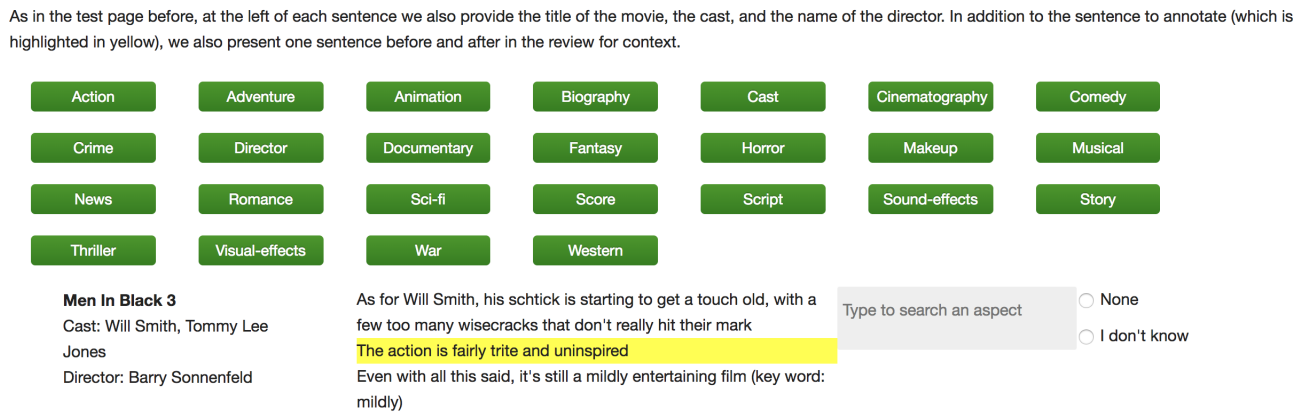


FIGURE 5.1: Web interface of the user study

an aspect). We ask each crowd-worker to annotate 10 sentences in total. We allow them to annotate sentences with "None" if they think that no aspect was discussed in the sentence or with "I don't know" if they are confused about the aspects that are discussed in the sentence. To enforce the accuracy of the annotation task we provide sufficient explanations regarding the semantics of each aspect. We present one sentence from the same movie before and after each sentence we ask crowd-workers to annotate. We also provide adequate contextual information as the title of the movie, the cast, and its director. We ask workers to read carefully each sentence and the explanation for each aspect before selecting the aspect(s) that are discussed in a sentence. We also provide examples of correctly annotated sentences to ensure that workers understand the task of sentence annotation.

5.3 Characterization of the human annotated sentences

This section analyzes the dataset resulting from the user study. We evaluate the agreement among the annotations of the three different workers for each sentence as well as the number of aspects per sentence.

Let us first define the notation we use for the analysis. We define a *label* as the set of aspects a worker selects for a sentence. An *annotation* refers to the aspect or set of aspects selected by the majority of workers. As mentioned in Section 5.2 we allow users to annotate a sentence with one or more aspects. We say that sentences for which the three workers annotate with the same labels have *full agreement*. These cases are simple, as there is no doubt on how to annotate the sentence. In some cases, however, there is only partial agreement, i.e, only two out of three workers agree. For example, a sentence annotated with `['Story', 'Story', 'Action']` defines a *partial agreement* where we use a *relaxed match* and annotate instead with the aspect that the majority of workers selected, in this case "Story". In some cases workers may annotate sentences with multiple aspects. For example, we consider a sentence annotated with `['Story', 'Story', {'Story', 'Action'}]` as partial agreement even though the aspect 'Story' appears in the three annotations. Again the aspect that the majority of workers agree upon is "Story". To strengthen reliability of our evaluation results, we discard sentences where the three workers disagree or sentences that two or more workers annotate with 'I don't know'.

Table 5.1 presents the number of sentences with full agreement, partial agreement, and no agreement among the three workers. The first column illustrates the

TABLE 5.1: Workers agreement for annotated sentences (exact match)

	All	Single aspect	Multiple aspects
Full agreement	218	215	3
Partial agreement	223	209	43
No agreement	159	99	31

TABLE 5.2: Workers agreement for annotated sentences (relaxed match)

	Number of sentences
Full agreement	268
Partial agreement	241
No agreement	91

agreement for all annotated sentences, the second column illustrates the agreement for sentences annotated with only one aspect, while the third column presents the number of sentences when at least one worker annotates sentences with several aspects. We separate the annotated sentences with one aspect and the ones with multiple aspects because the comparison is more direct for sentences annotated with one aspect, whereas for multiple aspects there is some ambiguity as aspects could partially overlap. We observe that workers annotate more sentences of our reference dataset with only one aspect rather than several aspects, we also observe that only in 36% of the cases workers fully agree on the same aspect or set of aspects. This result highlights that the task of understanding the aspects covered by a particular sentence is hard even for humans.

Finally, Table 5.2 presents the corresponding number of sentences when we use relaxed match to annotate sentences. For example, we consider a sentence annotated with `['Story', 'Story', {'Story', 'Action'}]` as a full agreement as the aspect 'Story' of the first two annotations is included of the third one. This relaxation increases to 268 the number of sentences with full agreement, and to 241 the number of sentences with partial agreement. Finally, we have 91 sentences having three different annotations. The rest of our analysis focuses on the 509 sentences for which the majority of workers agree.

Figure 5.2 shows a bar plot with the number of sentences annotated with each aspect. Workers annotate sentences with all aspects but with a skewed frequency distribution. Workers annotate 37 sentences with "I don't know". Also workers rarely select some aspects such as "News" or "Documentary", this is intuitive as these genre are unpopular in our corpus. Workers give the label "None" to many sentences. This result is expected as many sentences simply describe the plot or scenes of the movie, for example: "We see her open the envelope but never know what it said". These kinds of descriptive sentences are not relevant for aspect-based opinion summarization, which focus on aspects of movies. Thus, our algorithm should recognize that there is no aspect discussed in those sentences.

One of the most frequent aspects is 'Story' as workers systematically use it to annotate sentences describing the actual story of the movie rather than an opinion about the story of the movie. For example, three different workers consistently annotate the sentence "Her mother drives to this town to figure out the connection but the girl goes missing" with the aspect "Story". This reflects that some workers misunderstand the aspect "Story". We find no evidence of similar misunderstandings for other aspects.

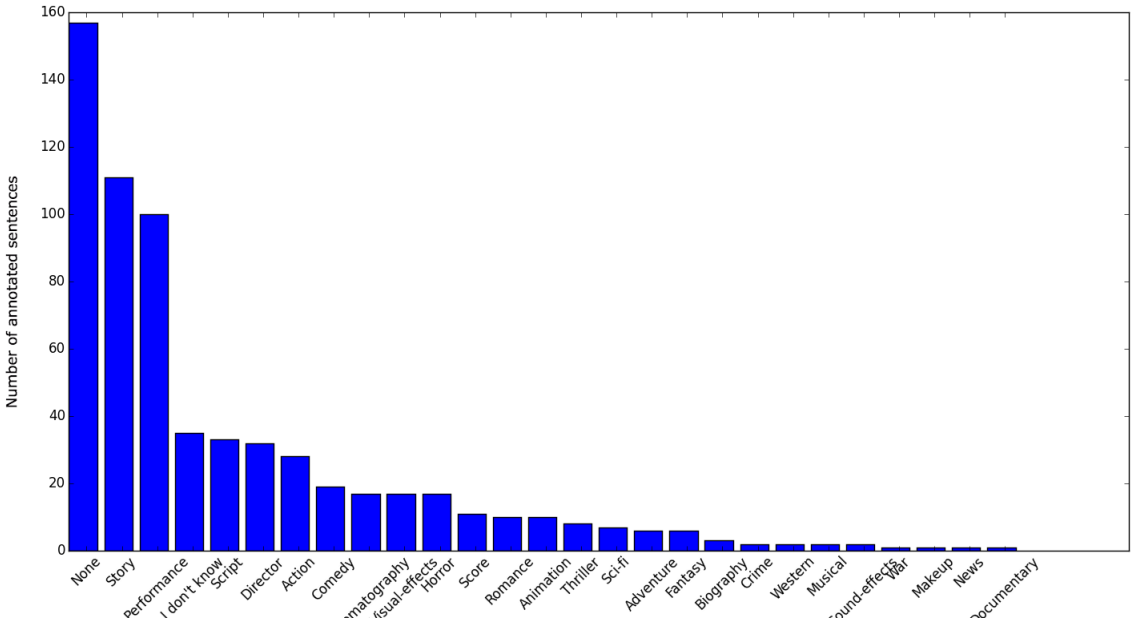


FIGURE 5.2: Distribution of aspects used to annotate sentences

Figure 5.3 shows the distribution of the number of aspects per sentences. We observe that the majority of labels contain just one aspect (1600 labels). Workers annotate some complex sentences (with length larger than 160 words) with up to five aspects.

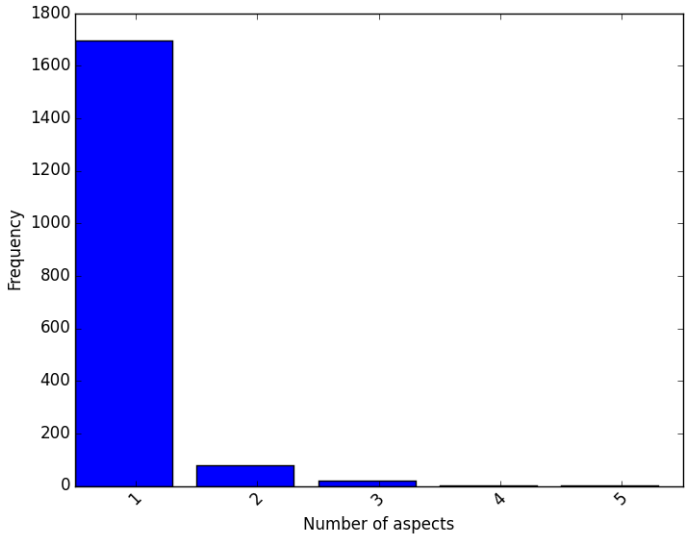


FIGURE 5.3: Distribution of the number of aspects per label

TABLE 5.3: Definitions of TP, FP, FN, and TN

	Exact match	Relaxed match
TP	$AN_u == AN_{alg}$	$\|AN_u \cap AN_{alg}\| >= 0.5 * \|AN_{alg}\|$
FP	$AN_u != AN_{alg}$ *and* $AN_{alg} != \text{“}none\text{”}$	$\|AN_{alg} \cap AN_u\| == 0$ *or* $\|AN_u \cap AN_{alg}\| < 0.5 * \|AN_{alg}\|$
FN	$AN_{alg} == \text{“}none\text{”}$ *and* $AN_u != \text{“}none\text{”}$	$AN_{alg} == \text{“}none\text{”}$ *and* $AN_u != \text{“}none\text{”}$
TN	$AN_{alg} == \text{“}none\text{”}$ *and* $AN_u == \text{“}none\text{”}$	$AN_{alg} == \text{“}none\text{”}$ *and* $AN_u == \text{“}none\text{”}$

5.4 Evaluation metrics

Our goal is to evaluate whether the annotations produced by the sentence annotation algorithm match the human annotations. We rely on standard metrics to evaluate classification methods: precision defined in Equation 5.1, recall (Equation 5.2), and F1-score (Equation 5.3). The definition of true positive and false positive in our case, however, depends on the choice of matching function between workers' annotation and the annotation of our algorithm. To this end we use two definitions to measure true positives and false positives summarized in Table 5.3.

The first definition uses an *exact matching* between the annotation given by our method, AN_{alg}, and the annotation given by workers, AN_u. If the two annotations match and they are different than 'none', then we consider a true positive. When both annotations are none, then we consider a true negative (i.e., there are no aspects discussed in the sentence). If the annotation is different than 'none' and the algorithm annotation is different than human annotation, we consider a false positive as the algorithm gave the wrong annotation. If workers give the annotation `['Story', 'Action']` to a sentence, whereas our algorithm produces the annotation `['Action']`, the exact match definition would consider this instance as a false positive, which is too strict as the annotation of the algorithm is partially correct.

The second definition relies on the *relaxed matching* (i.e., based on set membership) of sentence annotations, if the algorithm agrees with at least some of the labels then we would consider this as a true positive. In order to support comparisons between sets of annotations, we consider as true positives only annotations produced by our method that overlap more than 50% with the human annotations as shown in Table 5.3. For example, the annotation `['Action','Visual-effects']` generated by our method is considered as true positive w.r.t. the human annotation `['Story', 'Action']` because $|AN_u \cap AN_{alg}| = 1$.

Given the definitions in Table 5.3, we define Precision, Recall, F1-score as follows.

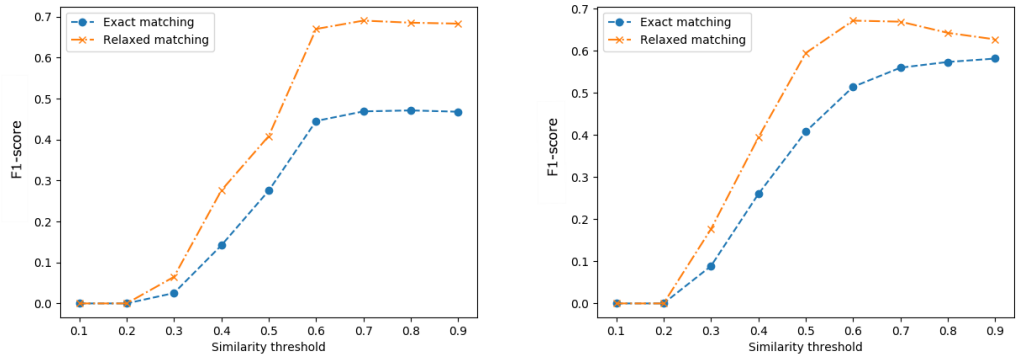
$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \tag{5.1}$$

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \tag{5.2}$$

$$F1 - score = \frac{2 * precision * recall}{precision + recall} \tag{5.3}$$

5.5 Parameter Tuning

In this section, we use the reference dataset with sentences that workers annotate manually to finalize our evaluation method and tune the parameters of the signature extraction and sentence annotation algorithms.



(A) Max-pairs metric, α=1 and d=0.15 (B) Max-count-pairs metric, α=1 and d=0.15

FIGURE 5.4: F1-score of max-pairs and max-count-pairs metrics on the reference dataset

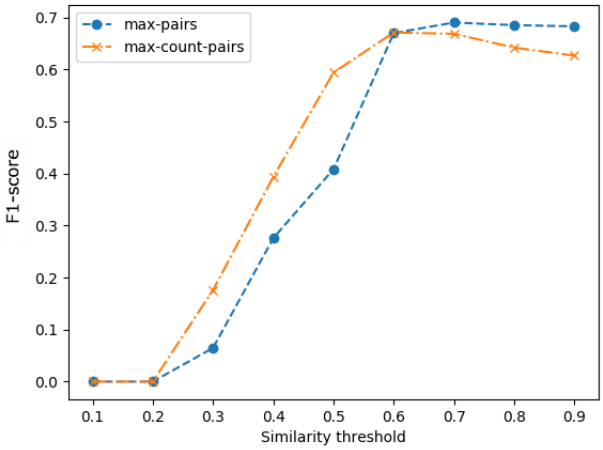
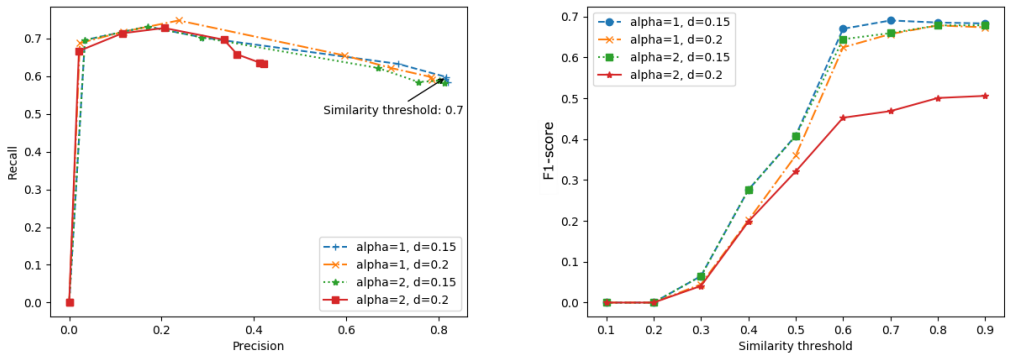


FIGURE 5.5: F1-scores of max-pairs and the count-max-pairs metrics using relaxed matching, α=1 and d=0.15

Selecting the matching method to compute precision, recall, and F1-score We study the impact of relying on the exact matching or the relaxed matching to compute the evaluation metrics introduced in the previous section. Figure 5.4 illustrates the F1-score of sentence annotation when varying the similarity threshold. Figure 5.4a relies on the max-pairs metric to compute the similarity between a sentence and an aspect signature, whereas Figure 5.4b relies on the max-count-pairs metrics. For these results, we set the signature extraction algorithm with $\alpha = 1$ and similarity distance d=0.15; our conclusions are similar for other settings. We see that comparing the annotations of our algorithm with human annotations using the relaxed matching leads to higher F1-scores when compared to the exact matching. For example, in Figures 5.4a we observe that for the max-pairs metric at θ=0.6, the F1-score using relaxed matching is 0.67 and for exact matching is 0.45. As discussed in the previous section, the exact matching can be too strict in cases when workers or the algorithm selects more than one aspect per annotation. In particular, we find that our algorithm sometimes gives more detailed annotations with multiple aspects, whereas workers pick fewer aspects. For example our algorithm annotates the sentence "It surpasses all expectations in action horror and comedy you'll love it so don't waste

time see it"with `['Comedy', 'Action', 'Horror']` while workers only annotate it with `['Action', 'Comedy']`. In another example, our algorithm annotates the sentence "Despite the strong acting of the leads and the stunning cinematography of Wyoming the central story felt rushed and incomplete" with `['Story', 'Cinematography']` while workers with `['Cinematography']`. Exact matching counts these examples as false positive, which is too strict. Therefore, the rest of the analysis relies on the relaxed matching.

Similarity metric for sentence annotation Section 5.1 discusses two possible similarity metrics to annotate sentences with aspects: Max-pairs and Max-count-pairs. Figure 5.5 evaluates which metric leads to higher accuracy. It presents the F1-score when varying the similarity threshold. Although Max-count-pairs leads to higher F1-scores for thresholds up to 0.6, Max-pairs has higher F1-scores for higher thresholds. The difference between the two curves for thresholds higher than 0.6 is small. To better understand the differences, we present the confusion matrices for thresholds 0.6, 0.7, 0.8, and 0.9 using Max-pairs in Table 5.4 and using Max-count-pairs in Table 5.5. Max-count-pairs results in more false negatives (158 versus 132 for the Max-pairs at 0.7) and in less true positives (185 versus 196 for Max-pairs at 0.7) but at the same time it has less false positives (16 versus 44 for Max-pairs at 0.7). Given the differences are small and the relatively small size of the dataset with manually-labelled sentences, we consider these results inconclusive. In the next section, we use both similarity metrics to annotate sentences in the entire IMDB dataset to evaluate which metric leads to a larger number of sentences annotated.



(A) Precision-recall curve. (B) F1-score.

FIGURE 5.6: Comparison of different configurations of the signature extraction algorithm

Selecting parameters of the signature extraction and sentence annotation algorithm The reference dataset allows us to go back to evaluate which configuration of the parameters (α and d) of the signature extraction algorithm from Chapter 4 as well as which similarity threshold (θ) for the sentence annotation algorithm lead to best precision/recall when annotating sentences. Figure 5.6a presents the precision-recall curve of sentence annotation using the Max-pairs metric for different parameters of the signature extraction algorithm (the trends are similar when using the Max-count-pairs metric) and Figure 5.6b presents the F1-score when varying the similarity threshold. We consider combinations with α=1 and 2, and d=0.15 and 0.2. Table 5.4 helps explain the shape of the precision-recall curve. If we use a small

similarity threshold we annotate sentences with more aspects, hence we reduce the number of false negatives while the number of true positives remains the same. As the similarity threshold increases, the precision value also increases until it slightly drops at a threshold of 0.7. With higher thresholds, our algorithm annotates more sentences with "None", which in turn increases the number of false negatives and decreases the number of true positives. We rely on these results to tune the parameters for the rest of this dissertation as follows.

- **Similarity threshold of sentence annotation, θ:** Given the shape of the precision-recall curve, we select θ=0.7, which gives a good balance between precision and recall and ensures that we get enough annotated sentences to pick from. Higher thresholds increase the number of sentences annotated with 'None'.
- **Signature extraction degree of overlap, α, and similarity distance, d:** As we can see in Figure 5.6b, the configuration α=2 and d=0.2 yields the lowest F1-scores across thresholds. When we increase the degree of overlap of terms between aspect signatures, our algorithm annotates sentences with more aspects, which reduces the number of false negatives while the number of true positives remains the same. Although the other three configurations lead to similar F1-scores, setting α =1 and d=0.15 leads to higher F1-score when θ=0.7. Hence, the rest of this dissertation will use α =1 and d=0.15.

TABLE 5.4: Confusion matrix of max-pairs metric using relaxed matching, α=1 and d=0.15

	Predicted No	**Predicted Yes**
Similarity threshold 0.6		
Actual No	TN=76	FP=16
Actual Yes	FN=56	TP=115
Similarity threshold 0.7		
Actual No	TN=121	FP=44
Actual Yes	FN= 132	TP=196
Similarity threshold 0.8		
Actual No	TN=123	FP=44
Actual Yes	FN= 133	TP=193
Similarity threshold 0.9		
Actual No	TN=123	FP=42
Actual Yes	FN= 136	TP=192

5.6 Evaluation Using Manually-labeled Sentences

We use the parameter settings identified in the previous section to evaluate the accuracy of our sentence annotation method when compared to two existing methods to annotate sentences with aspects given the aspect signatures: the *matching-coefficient* (Equation 3.3.2) and a simple count of matching terms i.e. *matching-count* [98]. The F1-score is similar for both baseline methods i.e. matching-coefficient and matching-count which is 0.68. Both methods have the same F1-score first because both methods are based on the exact match between the sentence and the terms. We observe that the recall is slightly lower than the recall given by the max-pairs similarity, while

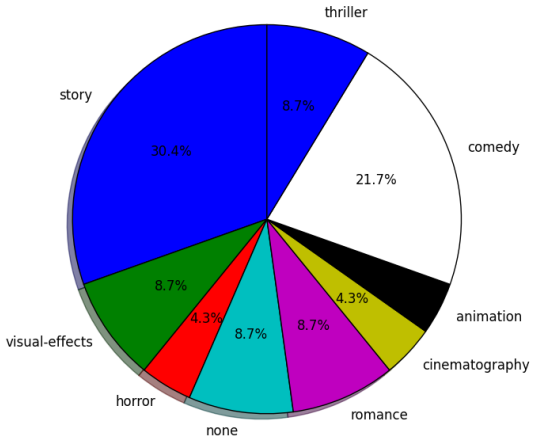
TABLE 5.5: Confusion matrix of max-count-pair metric using relaxed matching, α=1 and d=0.15

	Predicted No	**Predicted Yes**
Similarity threshold 0.6		
Actual No	TN=76	FP=16
Actual Yes	FN=56	TP=115
Similarity threshold 0.7		
Actual No	TN=125	FP=25
Actual Yes	FN= 158	TP=185
Similarity threshold 0.8		
Actual No	TN=125	FP=24
Actual Yes	FN= 170	TP=174
Similarity threshold 0.9		
Actual No	TN=127	FP=18
Actual Yes	FN= 181	TP=167

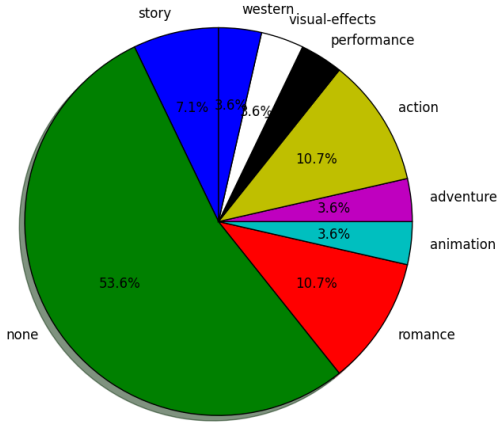
the precision is the same. The matching-coefficient and matching-count methods require an exact match with the aspect signature which leads to fewer true positives. Our method on the other hand, relies on the similarity between aspect signature and sentences, which increases the true positives while maintaining a small number of false positives. For example both methods: the matching-coefficient and count of matching terms annotate the sentence "But Tobey proved that he is not only a great actor but can also be an action star" only with {"Action"}, due to the presence of the word "action" in the sentence, while using the max-pairs similarity metric it was annotated by {"Action, Cast"}.

Our manual inspection shows that the max-pairs method annotates more sentences with "None" than workers. To better understand why, we analyze the distribution of aspects workers give for sentences that our method annotates with "None" (see Figure 5.7a). On the other hand Figure 5.7b presents the distribution of aspects our method gives for sentences that workers annotate with "None". First we find that 53.6% of `'None'` annotations provided by workers is also returned by our annotation method (see Figure 5.7b). Second we observe that our method annotates only 7.1% of sentences with "Story" compared to 30.4% for workers. This explains the problem we present in Section 5.3 which is that workers annotate sentences that discuss the story of the movie rather than sentences describing the quality of the aspect "Story". Third we find that workers annotate 21.7% of sentences with 'Comedy' compared to 0% for our algorithm. This shows that our algorithm is unable to understand nuanced sentences talking about "Comedy". In this case `'None'` is usually the annotation of complex sentences. For example, our method annotates the sentence "You believe that they are flying on brooms" or "It just looks hideously fake" with `'None'`, workers annotate them with `'Visual-effects'`.

We also observe other divergences where the algorithm and humans give different aspects, we study these cases and discuss the results. First we find that our method annotates sentences with aspects that are semantically related to the annotations of workers. For example, our method annotates the sentence "The Hunger Games has plenty of tension and suspense" with `['Action']` compared to `['Thriller']`. Similarly, our method annotates the sentence "It always helps if you can back that tension up with something truly unpleasant" with `['Action']`



(A) Distribution of workers aspects for sentences annotated by our method with "None"



(B) Distribution of aspects annotation by our method for sentences that workers annotate with "None".

FIGURE 5.7: Contrasting the 'None" annotations produced by our method against workers

compared to `['Horror']`. Second our method annotates sentences correctly despite the fact that workers annotate them with `'None'`. For instance, workers annotate the sentence "I love seeing him in the smartest person in the world roles although I was corrected today saying Sherlock's brother Mycroft Holmes Stephen Fry is actually smarter" with `'None'` and our method annotates it with `['Cast']`.

To summarize, after manual inspection we distinguish three cases where our method generates annotations that diverge from those given by workers (in a total of 89 sentences).

- When our method is unable to infer aspects that are not explicitly mentioned in the sentence, so it annotates it with `'None'` (51% of the cases).
- When we annotate sentences with specific aspects (e.g., `'Cast'`) although workers annotate them with `'None'` (e.g., when actors names are not recognized by workers) (26% of the cases).

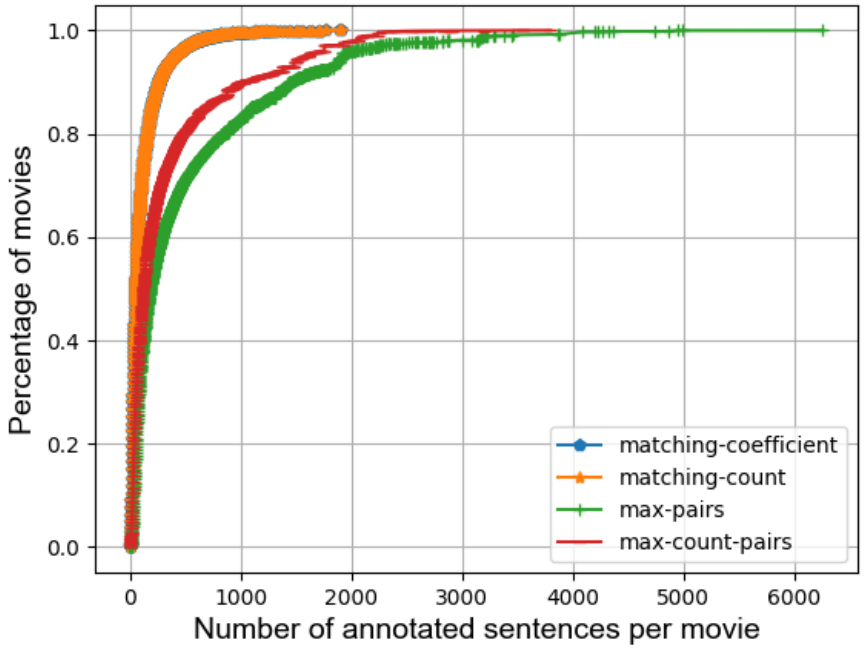


FIGURE 5.8: Cumulative distribution of the number of annotated sentences using different methods

- When annotations generated by our method are semantically related to those produced by workers (23% of the cases).

The first case where we label sentences with 'None' will reduce the set of sentences that our summarization algorithm will have to pick from, which can limit the richness of the summaries. But it will not introduce errors where we include sentences of aspects that are not relevant to the user. In the second case, the algorithm identifies more sentences and it should be better for summarization, whereas in the last case the output of the algorithm is equivalent to that of human workers.

Finally, we characterize the divergent sentences, where workers fail to agree and we assume are more difficult to annotate. Our method agrees with at least one worker in 91% of the sentences. Which means that even in these harder cases, our method agrees with at least one worker in the vast majority of cases.

5.7 Large-scale Evaluation

Our evaluation so far has focused on a small fraction of sentences for which we obtain manual annotations from human workers. In this section, we broaden our evaluation to the whole corpus. Although we have no manual annotations for the entire set of sentences, this larger evaluation will help us understand how the sentence annotation method behaves across different types of movies (from blockbusters to more niche films).

Figure 5.8 shows the cumulative distribution of the number of annotated sentences per movie when using different similarity metrics. This figure shows that the max-pairs similarity annotates more sentences than the max-count-pairs, which in turn annotates more sentences than both baseline methods: the matching-coefficient

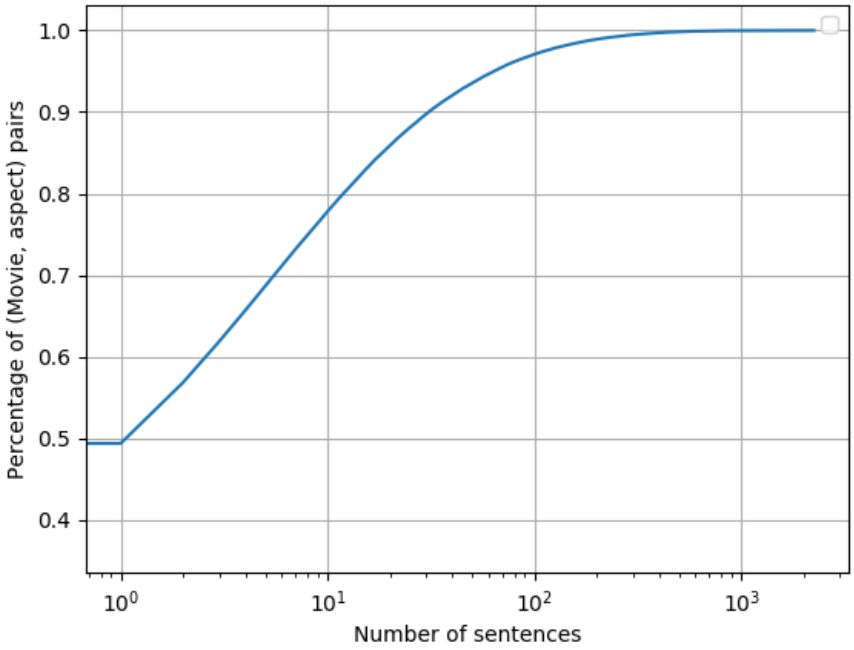


FIGURE 5.9: Cumulative distribution of the number of sentences per movie, aspect pairs

and the matching-count. We also observe that both baseline methods annotate the same number of sentences. These results are in line with our findings in Section 5.5, which showed that the baseline methods is unable sometimes to annotate sentences with the right aspect because it requires an exact match rather than similar terms to the aspect signature. We have seen that max-count-pairs achieves higher recall as it has a larger number of false negatives, which means that it often fails to annotate sentences with aspects. Given that we want to have a rich set of the sentences to pick from to generate summaries, we select the max-pairs similarity distance to annotate sentences.

Figure 5.9 shows the cumulative distribution of the number of annotated sentences per movie-aspect pair. We observe that for 80% of the movie-aspect pairs the number of annotated sentences is less than 10. This is due to the variety of the number of sentences per movie category and to the wide difference between the number of annotated sentences per aspect. Figure 5.10 presents the cumulative distribution of the number of annotated sentences for unpopular, popular, and blockbuster movies. As expected, unpopular movies have less annotated sentences than popular movies, which have less annotated sentences than blockbuster movies. 50% of unpopular movies have less than 100 sentences annotated, whereas blockbuster movie have at least 200 sentences annotated. The unpopular movie with the maximum number of annotated sentences has 300 annotated sentences, whereas for blockbuster movies the maximum number is 6200 annotated sentences. Unfortunately, some unpopular movies have only 8 annotated sentences. It is clearly a challenge to create a meaningful summary for movies with such a small number of sentences. Figure 5.11 shows the histogram of the number of sentences annotated per aspect across all movies in our dataset. For example the most popular Oscar category in the dataset is the aspect "Story", more than 700000 sentences are annotated with the

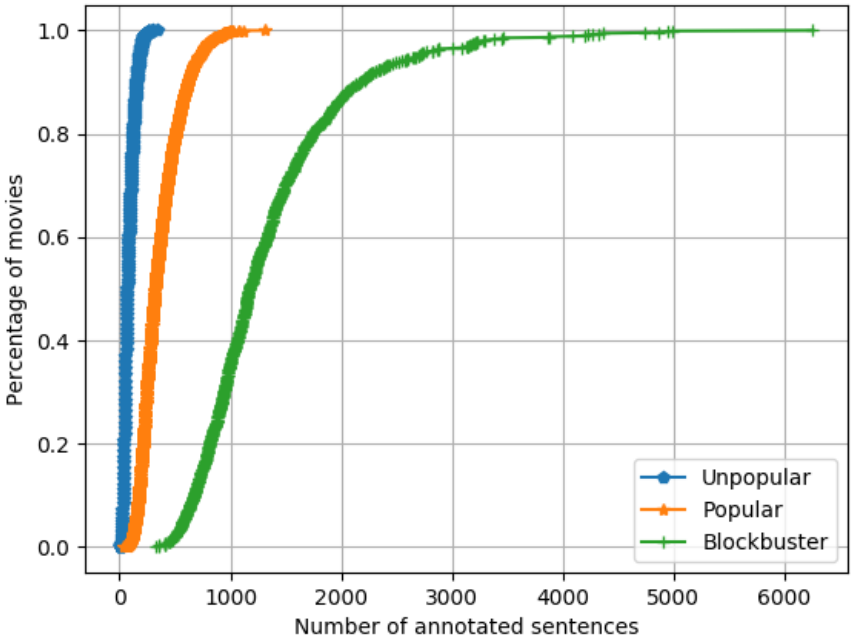


FIGURE 5.10: CDF of the number of annotated sentences for different movie categories

aspect "Story". This result is intuitive because in all movies we find comments about the story of the movie. On the other hand, the aspect "Makeup" is much less popular as it is a more specific aspect that is not popular across movies. Similarly, the genre "Action" is the most popular genre across movies. This result is also intuitive, as their are many movie genres like "Comedy", "Romance" or "Adventure" that contain action scenes. The imbalance in the number of annotated sentences per aspect is not necessarily an issue for our summary generation approach as we must select just a few sentences per aspect to include in the summary. One issue is if the user is only interested in niche aspects such as biography, documentary, or makeup as for many movies, there will be no reviews discussing these aspects and hence no sentences to pick from. A summary generation method that is too personalized may generate an empty summary for this user for many movies. We discuss how to address this issue in our summary generation method presented in the next chapter.

5.8 Opinionated sentences detection

In this section we explain how we select sentences that describe aspects with opinions. Take as an example these two sentences that are annotated with the aspect "Story": { "So here's the story first", "The story was funny through out" } the first sentence will not bring any value if we add it to the summary because it does not describe the aspect "Story", on the other hand the second sentence is valuable as it contains an opinion word "Funny" that describes the story of the movie.

We use part of speech tagging (POS) and lexical rules as summarized in Table 5.6 to detect opinion terms that describe the aspect terms [54]. For example we consider the sentence "This movie has an awesome cast" as an opinionated sentence because it matches the first rule in Table 5.6, the Noun "cast" (NN) being the aspect

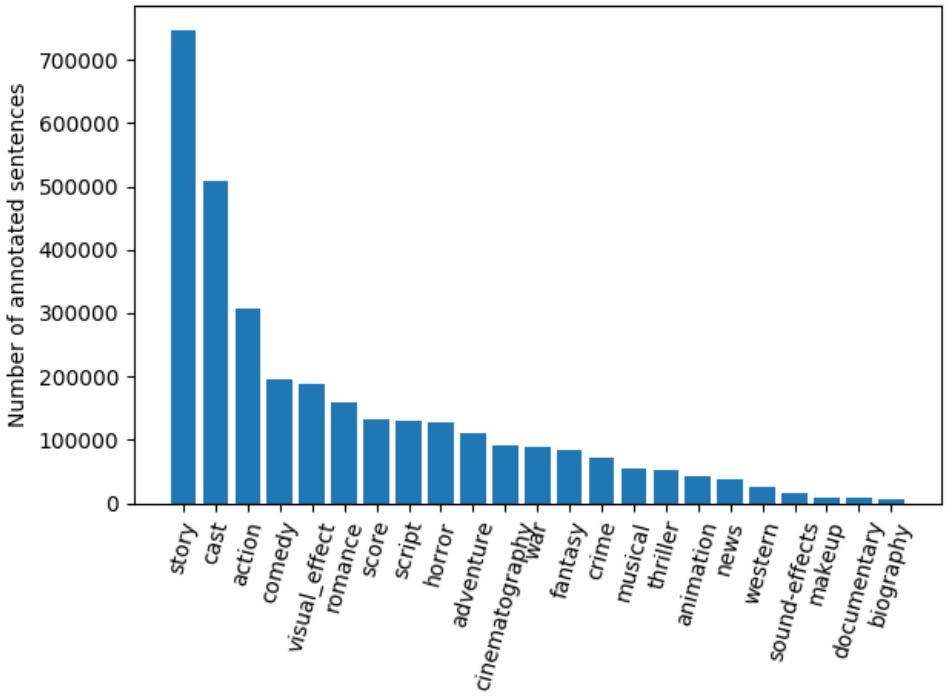


FIGURE 5.11: Histogram of the number of annotated sentences per aspect

and the adjective "awesome" (JJ) being the opinion term. This method was already used [37, 36]. The advantage of this method is that it is simple to use and does not require a deep semantic parsing technique that is computationally heavy. The main downside of this method is that we can have some false negatives, for example the sentence "The story of the movie itself, is really rather funny" won't be selected as an opinionated sentence, because the distance between the adjective "funny" and the aspect "Story", and the heuristic rules we use are at a small distance from the aspect word. The second disadvantage of this method is that it may detect sentences that are questions without answers, for example, "Is this really a funny story worth all this hype?". If this sentences is selected to be part of the summary it will not add valuable information because it does not describe the story of the movie. However, our method cannot deal with it correctly, because the possible aspect-opinion pair "funny-story" can be matched by one of our heuristic rules in Table 5.6 which is "JJ-NN ". Context information should be used to solve the problem. The last problem is that it does not handle irony, for example the sentence "My theory is that more than 70% of theater audiences fell asleep during the first 10 minutes and dreamed that they saw a good story". To solve this problem correctly, we would need a sophisticated algorithm that is able to detect ironic sentences, so that we would not consider them while generating our summary.

5.9 Conclusion

In this chapter we solve the problem of annotating sentences with aspects. Our method is based on the similarity between the aspect signatures and the noun terms in the sentence. We conduct a user study to first tune the aspect signature algorithm and the sentence annotation parameters. Then to compare the sentence annotation

TABLE 5.6: Examples of opinion words detection rules

Opinion terms detection rules
JJ - NN
NN - JJ
NN - VB - RB - JJ
NN - RB - RB - JJ
NN - VB - RB - RB - JJ

method with the baseline methods, we use the Prolific crowd sourcing platform and we annotate each sentence from 600 different sentences by three workers. We find that an α value of 1 and a similarity distance d of 0.15 and a similarity threshold θ of 0.7 give the best precision/recall trade-off (0.8/0.6). Our reference dataset evaluation is not conclusive enough to identify the best similarity as both similarity metrics (max-pairs and max-count-pairs) have similar precision/recall results. For this reason we did a large scale characterization over the entire dataset to evaluate the number of annotated sentences per similarity metric. We show that the max-pairs similarity metric outperforms the max-count-similarity metric as well as the baseline methods as it allowed us to have 61% more annotated sentences to pick from for generating the final summary. The results also show that there is a wide difference between the number of annotated sentences per movie category, in the worst case scenario we will have eight sentences annotated to pick from to generate our summary. In the next chapter we use the tuned parameters of the aspect signature algorithm and the sentence annotation method in order to generate the personalized aspect based summary.

Chapter 6

Personalized Summary Generation

This chapter presents our method to generate personalized, aspect-based summaries of movie reviews. Online reviews contain opinions about movies and movie aspects. The goal of our summarization is to capture the main opinions about aspects in the reviews. Aspect-based summarization is in contrast with traditional text summarization which selects a subset or rewrite some of the original sentences to capture the main points. Existing methods of aspect-based summarization [37, 55] generate summaries using a notion of importance based on features of the text only, independent of the reader. This lack of personalization is a weakness as different people often want different content for their summaries [87]. Marcu et al. [65] found that four different human judges select excerpts from a given text with only 25% overlap. Similarly, Salton et al. [89] found that the most important 20 paragraphs selected by 2 subjects have only 46% overlap. These results show that each person has a different perspective on the same text and highlight the need of personalization of summaries. In this dissertation, we achieve personalization by ensuring that summaries cover the aspects users are interested in.

Our method is extractive, i.e. it selects a subset of sentences from all of the reviews of a movie. We take as input the set of sentences annotated with aspects using the methods described in the previous chapters to generate a summary that covers the user interest together with the main opinions in the reviews. We formulate an optimization problem to generate summaries that maximize the coverage of the aspects the user is interested in and the representativeness of sentences. We start this chapter with a formal definition of this optimization problem (Section 6.1). Then, Section 6.2 describes our method to compute representativeness. Section 6.3 presents how we use CPLEX to solve the optimization problem. Sections 6.4, present the user studies we conduct to evaluate the quality of the ranking method and of the generated summaries.

6.1 Personalized summary generation: Problem definition

Given a set of opinionated sentences for a movie m, $O_m = \bigcup_{i=1}^{n} O_{a_i}$, where each O_{a_i} is the set of opinionated sentences annotated with aspect a_i. Our goal is to generate a personalized summary, $Z \subset O_m$, which satisfies the following properties:

1. Personalization: The summary should contain sentences that are relevant to the user. Previous work [22, 21, 5, 73] rely on a user profile defined as a vector of weighted keywords/aspects. Such an approach means that a user should manually input a weight for each (25) aspect which can be complicated and counter-intuitive for users. Instead, we ask users to pick the aspects they care about. We model the user profile as a boolean vector $U = \{u_{a1}, ..., u_{an}\}$, where

u_{a_i} is a boolean value that indicates whether the user is interested in a particular aspect a_i or not. Our final summary, Z, should maximize the coverage of the aspects the user is interested in.

2. Representativeness: Our summary captures the "representative" opinions about each aspect. We define a function, f, that takes as input an opinionated sentence, o, and an aspect, a, and returns the representativeness of o to a. Intuitively, we want f to return the largest values for sentences that capture the most popular opinion about the aspect. Popularity alone is not sufficient to capture representativeness as there will be a large number of sentences that express the most popular opinion. Section 6.2 formally defines f.

3. Length: A summary should have a limited (readable) size. This length constraint is an input parameter of our algorithm, which can be chosen based on the visible part of the screen or directly by the user. To this regards we set a threshold on the maximum length of the summary, $len(Z) \leqslant \delta_l$.

4. Redundancy: The summary should avoid sentences that convey the same information to capture more opinions within the length constraint. We define the threshold on the similarity between any two sentences in the summary as δ_{sim} to minimize redundancy and to ensure compactness.

We formalize the personalized, aspect-based summary generation problem as the following optimization problem:

$$\max \left(\sum_{a \in A} x_a * u_a + \sum_{a \in A} \sum_{o \in O_a} f(o, a) \right) \tag{6.1}$$

$$\text{subject to} \quad \sum_{z \in Z} len(z) \leqslant \delta_l \tag{6.2}$$

$$\forall a \in A, \forall o_i, o_j \in O_a, \; sim_a(o_i, o_j) \leqslant \delta_{sim} \tag{6.3}$$

Where:

1. x_a is a boolean that captures whether aspect a is covered in at least one sentence in Z.

2. $f(o, a)$ captures the representativeness of the sentence o for the aspect a in the reviews of m. We formally define the function f in Section 6.2, but we use a normalized definition so that $0 <= f(o, a) <= 1$. This normalization makes it possible to combine the two terms in a single optimization objective function.

3. $sim_a(o_i, o_j)$ is the similarity between o_i and o_j with respect to the opinion about a. We use the cosine similarity (defined in Chapter 3) between the opinion terms that describe a in o_i and o_j.

4. δ_l and δ_{sim} are two adjustable parameters to control the maximum length of the summary and the amount of redundancy allowed in the summary.

Equation 6.1 maximizes the sum of two terms. We discuss each term independently before explaining how we combine them into a single objective function.

- **Maximizing coverage of aspects in the user profile.** In isolation, we can map the objective $max \left(\sum_{a \in A} x_a * u_a \right)$ to the integer linear programming formulation of the maximum coverage problem as we are maximizing the sum of the

covered aspects (captured by x_a) in the user profile (captured by u_a). A summary that only maximizes the coverage of aspects in the user profile runs the risk of missing the most popular opinions in the reviews. In cases of mismatch between the user profile and the content of the movie reviews, the summary may also be meaningless. For example, if the user profile contains only the aspect "Action", but the movie is a documentary, the summary will likely be completely empty. In general, if there is space to add sentences to the summary and there are no more comments about the aspects in the user profile, it is better to capture representative sentences from the reviews.

- **Maximizing representativeness.** The objective $max\left(\sum_{a\in A}\sum_{o\in O_a} f(o,a)\right)$ maximizes the total representativeness of the sentences selected for the summary. This objective ignores the aspects in the user profile. It will be maximized when we pick sentences with the largest values of $f(o,a)$ across all aspects, so it will focus on popular sentences of popular aspects. A summary that only maximizes representativeness may miss opinions about the aspects the user is interested in.

- **Maximizing coverage and representativeness (Equation 6.1)** Our goal is to create summaries that cover the aspects in the user profile and capture representative sentences. We achieve this by adding these two objectives in Equation 6.1. Take as an example a sentence, o_p, that covers aspect a in the user profile (i.e., $u_a = 1$), where a is not covered in Z. We have $x_a * u_a = 1$ for o_p and remember that $0 < f(o_p, a) < 1$. The score of o_p is then the sum of the two terms, which will be between 1 and 2. Now say another sentence, o_q discusses none of the aspects in the user profile, it will have 0 in the first term and hence a total score lower than 1. To maximize the objective function, then we will pick the sentences with the highest scores (in this example o_p). Note that sentences covering aspects of the user profile (not yet covered in the summary) will always have highest scores, so the optimization will start by ensuring the coverage of aspects in the user profile. Say we have two sentences that cover a in the profile, o_{p1} and o_{p2}, and that $f(o_{p1}, a) > f(o_{p2}, a)$. To maximize the objective function, we must pick o_{p1}, which will have the highest score. We see that representativeness is a secondary objective. A sentence that covers no aspect in user profile will only be considered for the summary after we have already included sentences that cover all aspects in the user profile and there is still space left in the summary (i.e., $\sum_{z\in Z} len(z) < \delta_l$).

6.2 Representativeness

In this section, we define the representativeness function, $f(o,a)$. Our goal is that for a given aspect a this representativeness function returns the highest values for sentences that capture the most popular opinion about a. For example, assuming we have ten sentences from the reviews describing the aspect "Story" as "excellent" and one describing it as "bad", our goal is that the sentences describing story as excellent should have a higher representativeness value.

One challenge to identify popular opinions is that people often use a diverse vocabulary to express one single opinion. For example, if the aspect "Story" is described in five sentences with the opinion word "nice", in four sentences by "imaginative", and in other three sentences by the opinion term "creative", a simple frequency count would rank the sentences containing the opinion term "nice" higher

as it appears five times, whereas we can see that the major opinion about the aspect "Story" is that it is creative. To address this issue, we leverage the word embedding to cluster the opinion terms to identify similar opinions. This section first describes our method to cluster opinion terms and then formally defines f.

6.2.1 Clustering of opinion terms

We identify groups of sentences that give similar opinions about an aspect. One approach would be to apply a clustering algorithm on the embedding of all the opinion words of each aspect of each movie separately. This approach has two drawbacks. First, the number of data points for some movie-aspect pairs can be fairly small. Second, running a clustering algorithm for every movie-aspect pair is computationally heavy. Instead, we consider all opinion terms that appear in reviews of all movies and aspects. This approach is more efficient as we compute clusters once, plus it aggregates data across all movies and aspects, which increases the number of data points and hence the quality of the resulting clusters.

Our method clusters positive and negative opinions separately, because of the limitation of using word embedding, which assigns a high degree of similarity to antonyms as well as synonyms. For example when we cluster all opinion terms we find both opinion terms "good" and "bad" in the same cluster. We first select all the opinion words that appeared in the reviews of all movies, which we detect using the rules defined in Section 5.8. Then, we label each opinion word with its polarity using a compiled list of positive and negative opinion words widely used in nlp literature [38]. This compiled list is comprehensive and contains 1369 positive and 802 negative opinion terms, but it misses 94 opinion terms we find in movie reviews (for example, terms like competent or low-budget). We present the full list of added opinion terms in Table A.3 in the Appendix A. In addition, our manual inspection of the list identify three cases where the opinion polarities are incorrect in the context of movie reviews. For example "unpredictable" is usually a negative opinion term, but in our case it's the opposite, for example "unpredictable story" is a positive opinion about the aspect story. The same is valid for opinion term "funny" that is considered as a negative opinion in the compiled list. For this reasons we manually correct these errors and add the polarities of the opinion terms that were absent from the compiled list.

We then apply the K-means algorithm [62] to cluster opinion terms in the list of positive and negative opinion terms separately. We chose K-means because it is simple, unsupervised, and it finds non-overlapping clusters. K-means has only one parameter: the number of clusters, K. A small number of clusters may group different opinions in the same cluster, and a large number of clusters may result in having different clusters with the same opinion. To select the best number of clusters, we use the Elbow method [6]. Figure 6.1 presents in the x axis the number of clusters and the y axis the sum of squared errors (SSE) for each cluster in the set of positive and negative opinion terms. Using the elbow method we find that 89 is the best number of clusters for positive adjectives and 110 for negative adjectives. We denote the resulting set of opinion clusters as $C = C_p \cup C_n = \{c_1, ..., c_t\}$, where C_p is the set of positive opinion clusters and C_n the set of negative opinion clusters.

Table 6.1 and Table 6.2 show examples of the positive and negative clusters, respectively. We present the full list of positive and negative clusters in Tables A.4 and A.5 in the Appendix A. Our manual inspection of clusters shows that all the terms in each cluster are semantically similar, whereas terms in different clusters are distinct.

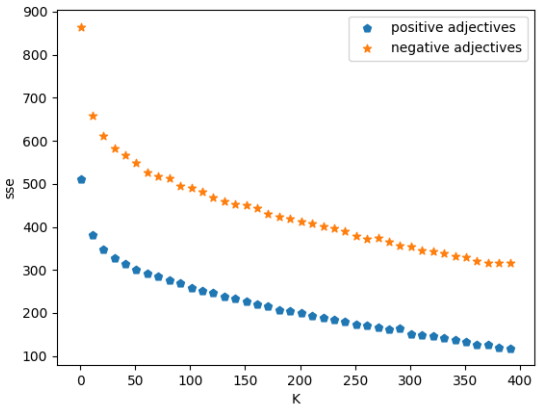


FIGURE 6.1: Sum of squared errors for different values of K while clustering the positive and negative opinions

TABLE 6.1: Example of positive adjectives clusters

Positive adjectives
[world-famous, well-received, prolific, well-known, adulatory, popular, well-regarded, well-established, better-known, prominent]
[innovative, visionary, unique, creative, bold, revolutionary, fresh, daring, ingenious, imaginative, inventive]
[top-quality, first-class, first-rate, pro, topnotch, strong, sturdy, top-notch, stellar]
[comic, amusing, comical, humorous, comedic, humourous, funny, hilarious, light-hearted]

6.2.2 Representativeness scoring function

This section defines the representativeness function, $f(o, a)$. Given two sentences, o_i and o_j, for a given aspect a, our goal is that $f(o_i, a) > f(o_j, a)$ if the opinion about a in o_i is more popular than that in o_j. We rely on the clusters of opinion terms to identify sentences with similar opinions about a given aspect. We split O_a into groups as $\{G_{a,c_1}, ..., G_{a,c_t}\}$ where each group of sentences $G_{a,c}$ contains sentences with opinion terms from the same cluster c. Note that one sentence can belong to multiple groups. For example, the sentence "This story is consistent and funny" belongs to two different groups. To address this issue, we select o's group for an aspect a as $G_a^o = max_j G_{a,c_j} \ | \forall j, o \in G_{a,c_j}$. Using the groups, we can compute the popularity of the opinion of sentence o about aspect a as the number of sentences in the group, which we denote as $|G_a^o|$.

One approach would be to set $f(o, a) = |G_a^o|$. This approach has two issues. First, the formulation of the optimization problem in Section 6.1 requires that $0 \leq$

TABLE 6.2: Examples of negative adjectives clusters

Negative adjectives
[lousy, bad, horrible, awful, sucky, shoddy, lame, terrible, lazy, crappy, poor, crummy]
[fragile, vulnerable, frail, brittle]
[heartbreaking, tragic, sad, sorrowful]
[toxic, contagious, infectious]

TABLE 6.3: Example of the trade-off between the computation time and optimality while varying relative gap tolerance values

Relative gap tolerance	e-04	0.05	0.1	0.15	0.25	0.5	0.9
Running time (in seconds)	91.29	56.06	37.15	33.98	33.62	33.55	33.27
Best feasible solutions	7.647	7.635	7.635	7.310	7.310	7.310	7.310

$f \leq 1$. Thus, we normalize $|G_a^o|$ by the size of the largest group over all the aspects, $|max_{i,j} G_{a_i,c_j}|$. Second, f would return the same value for all sentences in a given group, whereas the goal is that we only pick one sentence per group to avoid redundant opinions in the summary. We introduce a boolean variable, $x_{G_a^o}$, to capture whether a sentence in G_a^o is already present in the summary Z. Finally, we give preference to short sentences as they convey the same opinion without consuming as much of the allocated length budget. Thus, f should be inversely proportional to len(o), the number of words in the sentence o. We formally define the function f as follows:

$$f(o,a) = \frac{|G_a^o|}{|max_{i,j} G_{a_i,c_j}| * len(o)} * x_{G_a^o}, \tag{6.4}$$

$$\text{where } G_a^o = max_j G_{a,c_j} \mid \forall j \text{ s. t. } o \in G_{a,c_j} \tag{6.5}$$

This function rewards short sentences (by normalizing by the size of o), as well as popular opinions about popular aspects (by using the size of $|G_a^o|$). It also avoids redundancy by selecting only one sentence per group.

6.3 Summarization algorithm

This section presents our method to solve the personalized, aspect-based summary generation optimization problem formalized in Equation 6.1. We rely on the CPLEX solver [16] to solve this optimization. CPLEX is a commercial solver designed to solve (among others) large scale (mixed integer) linear problems. We use CPLEX version 12.7.1 with an academic license, which give us unrestricted access to the solver. We code our algorithm in Python and execute it on an Intel Core i7, 16 GB of RAM and MAC OS operating system. We use the *docplex* library to link between CPLEX and Python.

CPLEX has a few configuration parameters. One critical parameter is the *relative gap tolerance*, which controls the trade-off between the optimality of the solution and the computation time. CPLEX estimates first the *best bound*, which means the best solution on all possible values of the objective function. The relative gap tolerance parameter stands for the difference between the best bound and the best feasible solution found, divided by the value of the best feasible solution. If the relative gap tolerance is equal to zero, then we can guarantee that we get the maximal objective function value. Hence, the smaller the gap, the better the solution we get. If the gap is non zero, this means we can get a better solution by letting CPLEX work longer on the problem. The values of the relative gap tolerance fall in the interval between 0 and 1, and the default value in CPLEX is 1e-04. For example, to instruct CPLEX to stop as soon as it has found a feasible integer solution proved to be within five percent of optimal, we need to set the relative gap tolerance to 0.05.

We evaluate the trade-off between the optimality of the solution and the computation time for the "Die Hard" movie, which contains 294 opinionated sentences, in

Table 6.3. The table presents the computation time and the values of the best feasible solution for different settings of relative gap tolerance parameter. With a relative gap tolerance of e-04 we find a solution to the objective function with value 7.647, but this solutions takes 91.29 seconds. With a higher relative gap tolerance of 0.9, on the other hand, the computation time reduces to 33.27 seconds, but the solution is sub-optimal (the value is 7.310). In this dissertation, we run the algorithms off-line so we can afford the higher computational time. We set the relative gap tolerance with the default value of e-04 to ensure optimality of the results. The integration of this algorithm in an operational system, however, will require new methods to solve this optimization problem as even if we choose a high relative gap tolerance, the computation time is still more than half a minute, which is too much for users to wait for the summary generation.

6.4 User study

This section presents the user studies we conduct to evaluate the representativeness function defined in Section 6.2.2 and the quality of the summaries we generate with the algorithm from the previous section. Our optimization problem balances personalization (by covering aspects the user is interested in) with representativeness (by giving higher scores to sentences that express popular opinions about popular aspects). As in Chapter 5, we recruit workers using the Prolific Academic platform. The only criteria we have chosen is English fluency to make sure that all workers correctly understand the tasks. We first show that workers prefer such mixed summaries to fully personalized summaries. Then, we evaluate the mixed summaries against the baseline method from literature LexRank [26]. LexRank is a random walk-based method proposed for generic summarization.

6.4.1 Evaluation of representativeness function

The representativeness function f gives a score to each sentence under two main assumptions:

1. The adjective clustering algorithm identifies sentences that discuss the same opinion about one movie aspect.

2. Within a group of sentences, we should reward shorter sentences.

Ideally, we should present users with all opinionated sentences from the reviews of a movie, so that users can rank the most representative sentences. Unfortunately, given the number of sentences, this exercise would be too complex even for the most dedicated workers. Hence, we design an experiment to evaluate these assumptions that require users to read a small number of sentences. We ask each worker to evaluate three groups of five sentences each. Each group contains five random sentences that our algorithm identifies as expressing the same opinion about one movie aspect (i.e., sentences in a given $G_{a,c}$). We ask workers to select the one sentence that best summarizes the group of five sentences, if the worker thinks all the sentences express the same opinion. Otherwise, we ask them to select the option that not all sentences are similar. Workers also have a free-form text box to explain their choice. Figure 6.2 shows a screen shot of the web page of the user study. If users select the option that not all sentences describe the same opinion, then this contradicts Assumption 1. If the sentences they select among the group of five is not the shortest, then this contradicts Assumption 2. We conduct this user study on November 26, 2018 and obtained

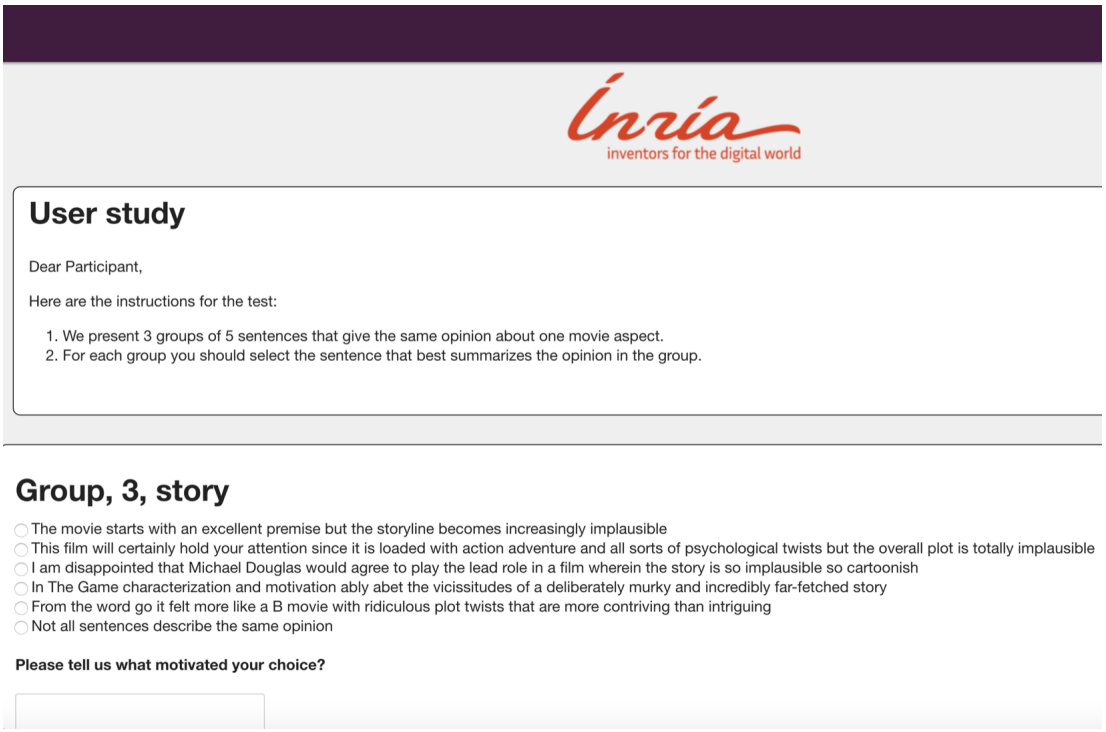


FIGURE 6.2: Screen shot of the questionnaire for evaluating the representativeness function

responses from 282 workers. In the beginning we present the instructions users have to follow to complete the task, than we present the three groups of five sentences.

Quality of adjective clusters We evaluate whether workers agree with our groups of similar opinions based on their answer to the question "Not all sentences describe the same opinion". Our results show that workers judge that 21% of our groups contain different opinions. This result is encouraging as workers agree with our grouping in the vast majority of cases.

Still, it is interesting to understand the reasons why workers identify some groups with different opinions. We read all the comments from users and identify four main issues that we summarize in Table 6.4. We observe that the majority of the responses point the issue of sentences' **contrastiveness**. Workers say the opinions are contradictory due to two main reasons. First, our method does not identify opinion negation in sentences. Workers give the example of these two sentences as contradictory "Actually not even the script, the plot is not good enough" and "The plot is very very good". To solve this problem we need to use lexical rules to detect the negation in sentences, then we use Wordnet [72] to associate the opinion negation with its antonym. For example, "not-good" should be associated with "bad". The second reason is that some sentences contain mixed sentiment about the same aspect. For example, the sentence "There's an interesting story but it doesn't unfold well" that belongs to the same group of the sentence "Well the film does have some nice cinematography as well as some interesting scenes and interesting premise". Users find these sentences different because the first sentences is not very positive.

The second reason is the difference in the **sentiment intensity**, for example workers find these two sentences different "The special-effects are decent but not stunning" and "The special-effects were very good but it did not make up for a poor

TABLE 6.4: Quotes from workers to explain why sentences in the groups express different opinions (From a total of 104 responses)

	# of Responses	Quotes from Workers
Contrastiveness	49	"These sentences describe conflicting opinions"
Multiple Aspects	19	"One is about the special effects, the other has both special effects and a muddled story"
Different Information	18	"choice 3 preferred another actor" "One opinion is just about one actor"
Sentiment Intensity	10	"Most of the sentences say that visual effects were OK (not great but not awful) but the 4th sentence says that visual effects were very good"

story". This is due to the fact that our grouping method does not take into consideration the superlative adverbs used to intensify opinions in sentences. One method to improve our clusters is to group sentences using downtoners (i.e. slightly, barely, somewhat) and intensifiers (i.e. very, extremely, really) separately.

Workers also find that the sentences are different when sentences cover **multiple aspects** or when sentences contain a **different information**, even if the opinion about the aspect we are evaluating is the same. For example, a worker finds these two sentences about the aspect "Cast" different "It's Sean Penn one of the best actors in the world in a good performance and Michael Douglas good as well" and "Michael Douglas as Nicholas is great and delivers a good performance". In fact when sentences are long they carry different information from each other, even if they describe the same aspect with the same opinion. We conjecture that a more direct question in the user study could have eliminated these issues. Instead of asking workers "For each group you should select the sentence that best summarizes the opinion in the group", we should have asked directly "For each group you should select the sentence that best summarizes the opinion about the aspect Story". Moreover, our scoring function focuses only on the opinion attached to the aspect, and ignores all other information that the sentence might carry. An alternative approach would take into consideration not only the representativeness of the opinion attached to the aspect, but the representativeness of all the words in the sentence.

Do workers prefer shorter sentences? For all cases where workers selected one of the sentences within a group, we evaluate the length of the sentences to verify Assumption 2. Figure 6.3 shows the cumulative distribution of both the length of sentences selected by workers (the orange line) and all the sentences in the experiment (the blue line). We observe that the length distribution of the selected sentences is a uniform sample from the length distribution of all sentences. This result contradicts

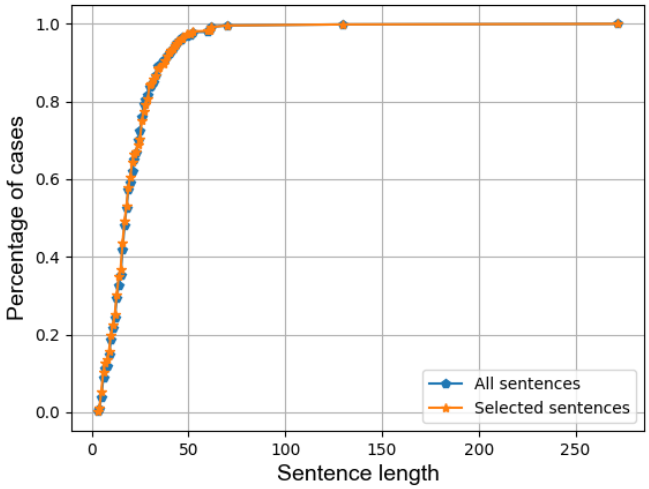


FIGURE 6.3: Cumulative distribution of the length of all sentences in the user study and the length of sentences selected by workers

our assumption: Workers present no clear preference to the shortest sentences. In order to understand the reasoning of workers for picking the sentences they selected, we analyze their textual responses. We find two divergent worker groups. The first group contains workers that prefer specific and detailed sentences, they gave comments such as "It's most in depth" or "This option tells the most information". The second group contains workers that prefer concise and simple sentences, they gave comments such as "It's quick and i know what to expect form the storyline" and "Short and on point". These results show that the amount of details users want to have in sentences varies from one user to another. Because of this divergence, we believe that the length parameter should be part of the user profile instead of being the same for all users. In our setting because we have a length constraint for the summary and because we only focus on the opinions attached to aspects, we believe that picking short sentences is a good strategy.

To summarize we observe that users take into consideration not only the opinion attached to the aspects but the information in the sentence as a whole. The preference of workers in terms of sentence length varies per worker.

6.4.2 Mixed versus fully-personalized summaries

In this section, we evaluate the level of personalization that users prefer. We compare the summaries generated using the optimization model presented in Equation 6.1, which we denote as *mixed* summaries, with summaries that contain only sentences that discuss aspects the user is interested, which we denote as *fully-personalized* summaries. We generate fully-personalized summaries using the following objective function:

$$max \left(\sum_{a \in A} x_a * u_a + \sum_{a \in A} \sum_{o \in O_a} f(o,a) * u_a \right). \tag{6.6}$$

The first term in this objective function aims to maximize the coverage of the aspects in the user profile, whereas the second term to maximize the representativeness of the selected sentences in the summary among sentences discussing the aspects the user is interested in (hence the multiplication by u_a).

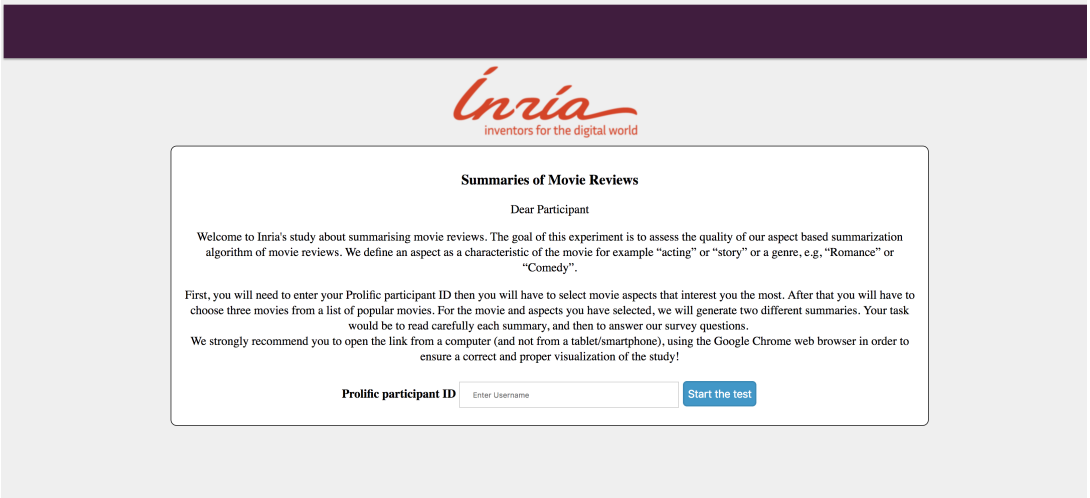


FIGURE 6.4: Screen shot of the welcome page of the user study

Experiment setup We design a user study to assess the usefulness and redundancy of each summary. Our goal is to shed light on whether users prefer more diversity in the summary in terms of aspects, or to have summaries that discuss only the aspects they are interested in. We ask workers to read the mixed and the fully-personalized summaries and to fill out a questionnaire assessing the summaries on several aspects related to its utility and quality. The questionnaire consisted of five key questions as follows:

- **Usefulness:** *From a scale of 1 to 5, how useful do you find summary 1/2* ? This can be positive/negative opinions about the movie or some critical facts about the movie. Score (1) - *Not useful at all*. Score (5) - *All the sentences are useful.*
- **Redundancy:** *From a scale of 1 to 5 how redundant do you find summary 1/2*? Score (1) - *All the sentences mean the same thing*; Score (5) - *The sentences are very unique, summarizing very different topics or issues.*
- **Summary review:** *What do you like most/least about the summary*? This is an open question where users give their opinion about what they liked or did not like about the summary. This question allows us to understand better what motivated the score they gave for the usefulness and for the redundancy.
- **Summary preference:** *Which summary do you prefer*? Users have to select one answer from four possible answers: Summary 1, Summary 2, Summary 1 and 2, and None.
- **Summary implementation:** *Would you use a movie reviewing website that implements the summary that you prefer*? Users can either answer yes or no.

The workers had no information about which method was used to generate the summaries, and to avoid bias of position, we randomly changed the order of the summaries on each questionnaire page. Figure 6.4 shows a screen shot of the web page of the user study where we present the instructions to follow to complete the user study. In the second page, we present a list of nine aspects and ask users to choose the aspects that interest them the most. We consider these aspects as the user profile. In the third page, we show 36 popular movies and ask users to choose exactly 3 movies that they have already seen. Figure 6.5 shows a screen shot of the

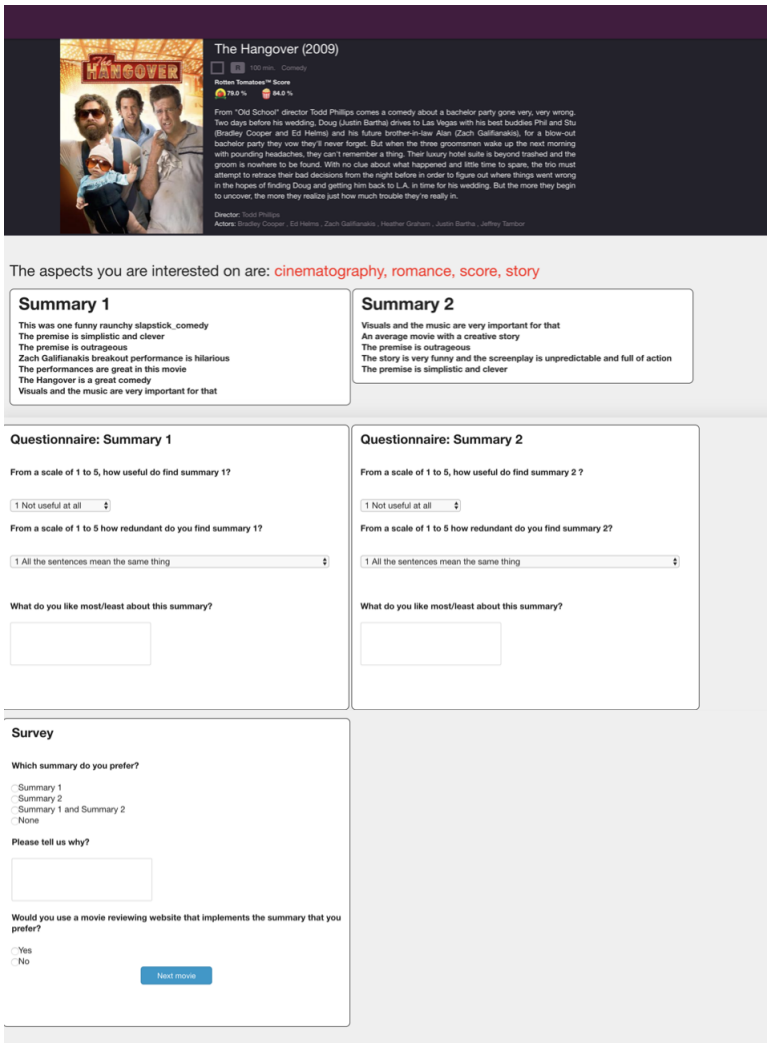


FIGURE 6.5: Screen shot of the questionnaire page of the user study

questionnaire page. On the top of the page we give the synopsis of the movie and the names of the cast. Below the movie description, we present both summaries, then we ask users to answer the questionnaire.

We conducted this user study on November 15, 2018, and 104 workers answered our questionnaire. In total we had 312 questionnaires filled.

TABLE 6.5: Workers responses to the question: "Would you like to see a website that implements the summary that you prefer"

	Mixed		**Fully-personalized**		**LexRank**	
	Yes	**No**	**Yes**	**No**	**Yes**	**No**
Second user study	106	84	63	12	-	-
Third user study	131	97	-	-	32	15

Workers prefer mixed summaries over fully-personalized summaries When answering to the question "Which summary do you prefer?", we have 132 answers

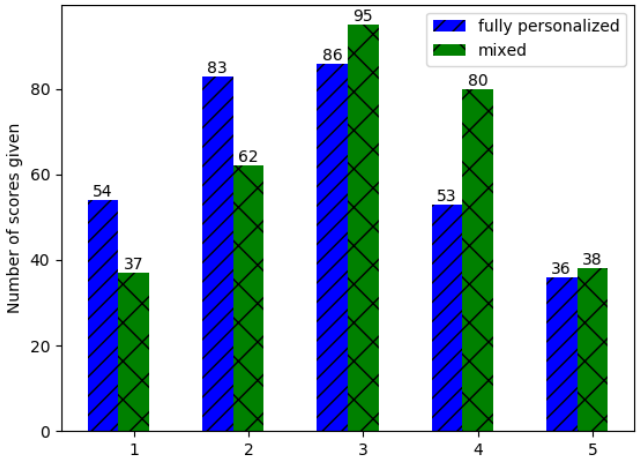


FIGURE 6.6: Histogram of the usefulness scores for both mixed and fully personalized summaries (From a total of 312 answers)

where workers declare that they prefer mixed summaries, compared to 78 where they declare they prefer the fully-personalized summaries. 65 worker said they don't like any summary and 36 liked both summaries. We evaluate the statistical significance of the results using the one sample-binomial significance that is a standard method to test preference data [91]. This test shows that workers preference for the mixed summaries is statistically significant with 95% confidence interval and a p-value of .00000001. Table 6.5 presents the statistics of workers' responses to the question: "Would you like to see a website that implements the summary that you prefer". Each row represents a different user study. 106 worker that preferred the mixed summaries said that they would like to see a website that implements it, and 63 worker that preferred the fully-personalized summaries said that they would like to see a website that implements it. We observe that workers who prefer the mixed summaries don't necessarily want to use a site that implements it. After analyzing the results we found that even for workers that gave low scores to the mixed summaries, they still preferred them over the fully-personalized one.

To understand the preference for the mixed summaries over the fully-personalized summaries, we analyze workers' free-text comments. We find that workers like that mixed summaries cover more aspects. Workers gave comments such as: "Is a good summary, it talks about several things and is fun to read" or "More detail about different aspects". Only three workers preferred the fully-personalized summaries because they only discuss the aspects they are interested in. These workers gave comments like "Was a little more personable and relatable " or "It gives a bit more focus on the cast". Another set of positive comments for the mixed summaries are about the fact that they are concise and less repetitive. Workers gave comments such as: "Summary 1 explains everything I wanted to know, this is achieved with short and clear sentences" and "Summary 1 encompassed a lot of aspects of the movie and was more detailed and unique.".

Figure 6.6 confirms this preference. It presents a histogram of the number of scores given to the usefulness for both summaries. We observe that workers give lower scores to the fully-personalized summaries: 99 responses have a usefulness score below 3 for the mixed summaries, compared to 137 for the fully-personalized

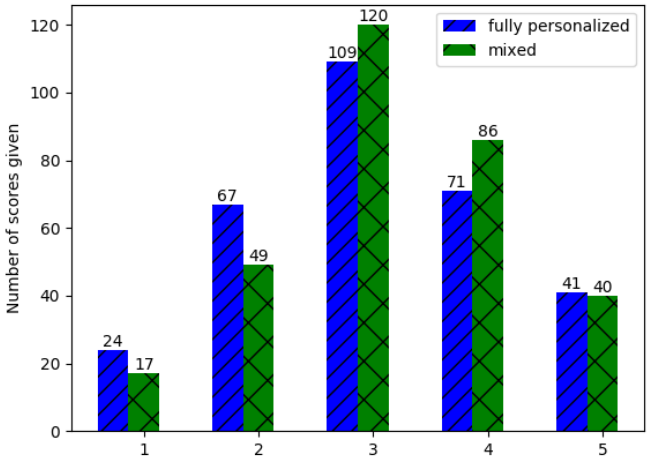


FIGURE 6.7: Histogram of the redundancy scores for both mixed and fully personalized summaries (From a total of 312 answers)

summaries. We also observe that 118 responses have a usefulness score above 3 for the mixed summaries, compared to 89 for the fully personalized summaries. To shed light on workers' complaints about the fully-personalized summaries, we analyze the textual comments for all responses with score less than 3 for both summaries. Some of the complaints are common between mixed and fully-personalized summaries. For example, users complain of bad grammar in both cases. We will study in more details the complaints about the mixed summary in the next section. Here we focus on the complaints that are unique to fully-personalized summaries. We find one major complaint about fully-personalized summaries: redundancy. Given that the fully-personalized summaries only cover the aspects the worker is interested in, many sentences will be discussing the same aspect even if with different opinions. Workers gave comments like "Its really redundant" or "Sentences only talk about performances.". Figure 6.7 that presents the histogram of the number of scores given to the redundancy for both summaries confirms this complaint. In fact 66 responses have a redundancy score below 3 for the mixed summaries, compared to 91 for the fully-personalized summaries. We also observe that 146 responses have a redundancy score above 3 for the mixed summaries, compared to 112 for the fully-personalized summaries. To understand if these results are statistically significant, we use the paired T-test, which is extensively used in literature to determine if two sets of ordinal data are significantly different from each other [91]. The p-values of the significance tests for usefulness and redundancy are statistically significant (the p-values are 0.0003 and 0.0498, respectively). This shows that workers find the mixed summaries more useful and less redundant than the fully-personalized summaries.

We conclude from this section that workers prefer the mixed summaries over the fully-personalized summaries. Therefore, we use the optimization function in Equation 6.1 to generate summaries to compare with the baseline method in the next section.

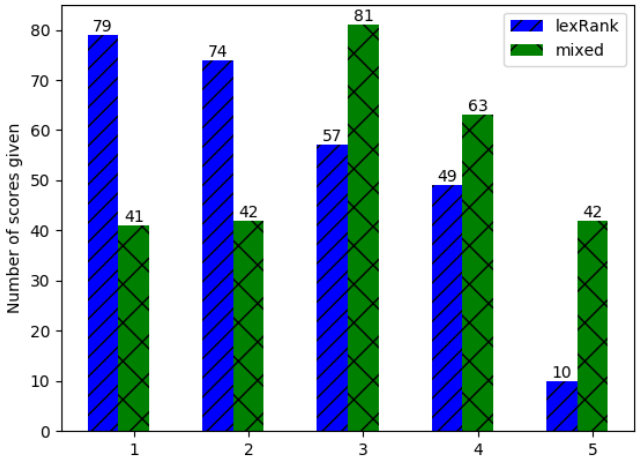


FIGURE 6.8: Histogram of the usefulness scores for both mixed and LexRank summaries (From a total of 269 answers)

6.4.3 Mixed versus baseline summaries

This section presents our final user study that compares the mixed summaries with LexRank [26], a widely used method to rank sentences according to their popularity in the original document. We generate the baseline summary by selecting the top sentences ranked using the baseline method until we reach the summary size.

Experiment setup Our user study setup is the same as described in Section 6.4.2. The only difference is that we replace the fully-personalized summary with the baseline summary. We conducted this user study on November 23, 2018 and 94 workers took the user study.

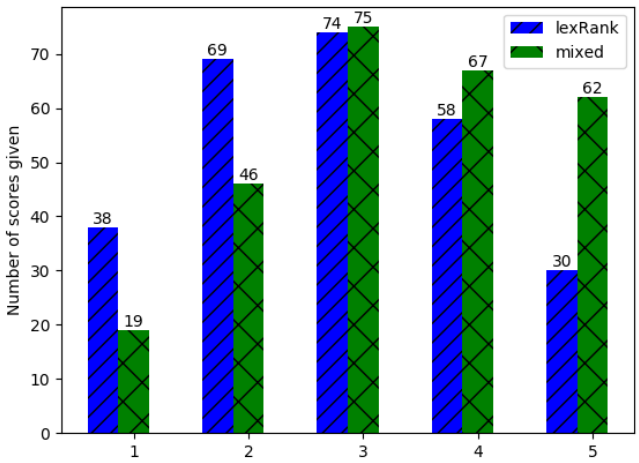


FIGURE 6.9: Histogram of the redundancy scores for both mixed and LexRank summaries (From a total of 269 answers)

TABLE 6.6: Table that summarizes users positive opinions about the LexRank summary (From a total of 48 answers)

	# of Responses	Quotes from Workers
Details	18	"It had enough detail" "The summary is very explanatory and simple"
Coherence	16	"Has good flow" "Tries to be a fluid text instead of bullet point"
Positivity	9	"Has good points about the movie"

TABLE 6.7: Table that summarizes users negative opinions about the LexRank summary (From a total of 137 answers)

	# of Responses	Quotes from Workers
Bad Grammar	55	"Very poorly written" "Grammar errors"
Lack of Details	35	"It's a shallow summary" "Kind of vague"
Repetitiveness	27	"Repetitive useless information" "Repetitive sentences"
Contrastiveness	13	"It seems to contradict" "Entirely contradict itself"

Workers prefer mixed summaries over LexRank summaries When explicitly asked to select the summary they prefer, 158 worker answered that they prefer the mixed summary, compared to 48 that declare they prefer the fully-personalized summaries. 49 of workers said they don't like any summary and 13 liked both summaries. The one sample-binomial significance test shows that workers preference for the mixed summaries is statistically significant with 95% confidence level and a p-value of .000001.

The more detailed questions about usefulness and redundancy of summaries confirm that the mixed summaries have overall higher scores than the LexRank summaries in both metrics. Figure 6.8 presents the histogram of usefulness, whereas Figure 6.9 presents that of redundancy. The paired T-test shows that the improvement over redundancy and usefulness of our method over LexRank is statistically significant with a p-value of 0.001.

We study the free-text comments left by workers who gave a positive score (a usefulness score strictly above 3) and those who gave negative scores (a usefulness score strictly below 3) to shed light on the reasons behind the scores. We read all the comments from workers and manually cluster the major praises and complaints

about the baseline method. Table 6.6 summarizes the positive responses and Table 6.7 summarizes the negative ones. Workers complain the most about the redundant information in the LexRank summaries and the fact that the summaries cover few aspects. On the other hand, workers like the fact that the sentences selected by the LexRank method contain more detailed information.

Table 6.8 shows the results of our manual clustering of the main praises about the mixed summary. We observe that the workers like most the **aspects coverage** in the summaries. Workers appreciate the fact that the mixed summaries cover multiple aspects, even if these aspects are not part of their user profile. This highlights the fact that users prefer to have a rounded summary that gives a better overview of the movie, as long as it includes the aspects they are interested in. The second aspect that the workers like about the summary is **conciseness**. Some workers like the fact the summaries are short, not repetitive and easy to digest. Surprisingly, the third most popular praise is the **positivity** of sentences. Some workers prefer to read positive comments, which they claim is encouraging to go see the movie. This may indicate some misunderstanding as some workers may have thought that the goal of the summary is to convince people to see the movie. Table 6.5 presents the statistics of workers' responses to the question: "Would you like to see a website that implements the summary that you prefer". We see that 131 workers that prefer mixed summaries would like to see them implemented in a website, as opposed to 32 for workers that prefer the baseline summaries.

TABLE 6.8: Table that summarizes users positive opinions about the mixed summary (From a total of 63 answers)

	# of Responses	Quotes from Workers
Aspect Coverage	34	"Covers nearly every aspect without any redundancy" "well rounded summary"
Conciseness	15	"It is simple and short, you get the most out of the least" "Says it all (almost), brief and self-explanatory."
Positivity	6	"I like that its positive" "Makes me want to watch it, very positive and emotionally driven review"

Negative opinions about mixed summaries Although workers prefer mixed summaries, the overall usefulness score of mixed summaries is still low (the average score is 3.09 for usefulness, and 3.38 for redundancy). We study the free-text comments made by workers who gave a score below 3 to understand the main issues with our current method and identify areas for improvement. Table 6.9 shows the results of our manual clustering of the main complaints about the mixed summary. It presents the number of sentences that contain each complaint, and two sentences quoted from the user reviews of the summary.

TABLE 6.9: Summary of negative opinions about the mixed summary (From a total of 44 answers)

	# of Responses	Quotes from Workers
Contrastiveness	15	"Many contradictions" "it's also incredibly contradictory"
Lack of Details	11	"Strong opinions but not enough details to support them" "Points are too short and unsubstantiated"
Bad Grammar	8	"It is really badly written" "The grammar is poor"
Lack of Cohesion	2	"Unrelated elements" "incomplete sentences that do not connect to each other"

We can see that users complain most about the **contrastiveness** in the summaries. They find it confusing to have both positive and negative opinions about the same aspect in the same summary. One way of dealing with this problem is to visually structure the summary by separating positive and negative sentences. Another solution would be to use colors to separate positive from negative opinions and to use the font size as an indicator of the representativeness of the opinions.

The second major complaint is the **lack of details**. Users find that the sentences are too short and do not explain the reasoning behind the given opinions. In Section 6.4.1, we show that some workers prefer more concise sentences. This disparity indicate that the length of sentences and the level of detail depends on the user. One interesting area for future research is how to incorporate this preference in the user profile.

The third most popular complaint is **bad grammar**. Our method does not evaluate whether sentences are grammaticality correct or not. This issue can be avoided by filtering out ill-formed sentences before running our optimization algorithm. A related issue and the final complaint is the **lack of coherence**. Our method is assembling sentences that come from different reviews, so there is no logical flow. Some workers dislike to see a summary as a list of bullets, they prefer to see a coherent paragraph instead. To address this issue, we should combine our extractive method with an abstractive method that would then form a coherent text from the set of sentences. Still, this is another controversial issue as some workers prefer to have a list, which they consider easier to digest.

The results of the user study are overall encouraging as workers largely prefer mixed summaries. Still, these results point out to a number of areas for improvement such as personalization according to more properties (length, format, and mixing positive and negative opinions) and better filtering of sentences (for instance, to avoid sentences with grammar mistakes).

6.5 Conclusion

In this chapter, we present the last two contributions of this dissertation. The first is our solution to the generation of aspect-based personalized summaries. We pose the problem as an optimization problem with an objective function that balances representativeness of opinions and coverage of aspects in the user profile. We leverage word embeddings to find groups of sentences with similar opinions about aspects. We can then compute the representativeness of sentences for an aspect based on the popularity of its group. To solve our optimization problem we use the state of the art CPLEX solver tool.

Our second contribution in this chapter is a user based evaluation of our summarization algorithm. In the first user study we discuss the representativeness function we propose. The goal of this study is to evaluate (1) the quality of our method for grouping similar sentences and (2) our assumption about picking the shortest sentence in the group of similar sentences. We find encouraging results as 79% of the workers agree with our groupings. In contradiction with our initial assumption, workers show no clear preference for short sentences. Some users claim to prefer conciseness and others details, which indicates that the level of detail in the summary should be part of the user profile. Our user studies then show that users prefer mixed summaries to both fully-personalized and LexRank summaries. The main issues users report with the fully-personalized summary is the redundancy as most of the summary only address the few aspects the users are interested in. Interestingly, one of the main complaint of the LexRank summaries (which picks sentences based on general opinion popularity with no personalization) is also about redundancy. This redundancy is due to the fact that LexRank is based on a popularity heuristic, so it only focuses on a few popular aspects. This type of summary uses the summary space limit without covering many aspects as opposed to the mixed summary.

The method we propose in this chapter is the first step toward studying the new problem of personalized aspect-based summary generation. The summaries we generate are not perfect yet. There is room for improvement and the user studies point to a number of promising directions that we will explore as a future work.

Chapter 7

Conclusion

The main motivation of this dissertation is to help users digest the vast amount of information available on online movie reviews. Users rely on the opinions expressed in reviews as source of information to make decisions about which movies to watch. Thus, a short summary of the main opinions in a large set of reviews should help them decide faster and better. This dissertation developed a set of novel algorithms and methods that together process the reviews of a movie into a personalized, aspect-based summary. This chapter first summarizes the main contributions of this dissertation and then it discusses future research.

7.1 Summary of contributions

This dissertation makes the following contributions:

An algorithm that extracts signatures of aspects The first contribution of this dissertation is an algorithm to extract aspect signatures from a corpus of reviews. We input a list with one seed term per aspect composed of movie genres and Oscar categories. Our algorithm then automatically identifies similar words to each seed term in the corpus of reviews and assigns them to the signature of that aspect. We rely on the state-of-the-art word embedding model Word2Vec to learn for each word its representation in a vector space. Word2Vec captures complex word relationships such as synonyms that are specific to the corpus used for training. The main advantage of our approach is that it is extensible, which means that if we have more data we can enrich the vocabulary of aspect signatures. In addition, the method is fairly simple as it makes use of no syntactic dependency parsing nor NLP-based heuristic. Our approach is different from current methods for aspect signature extraction in two ways. First our method simplifies the process of extracting signatures of aspects as it works in once step, we extract and group related terms to seed aspect terms at the same time. Previous methods [58, 93], on the other hand, rely on two different methods: one to extract the aspect terms and the second to cluster the similar terms in groups. Second, previous work pre-filtered only the high-frequency words as aspect candidates, whereas our method considers any word in the dataset as aspect candidate. As a result, our method is able to extract even unpopular terms such as compound terms or abbreviations.

A method that annotates sentences with aspects The second contribution of this dissertation is to develop a method that identifies the aspects discussed in the sentences and annotates sentences with the appropriate aspects. Annotating sentences with the right aspects is crucial, because we select sentences from the pool of annotated sentences in order to generate our final aspect-based, personalized summary.

We propose a novel method that we denote *max-pairs*, to annotate sentences with aspects that considers a similarity distance between the aspect signature and sentences. Our large-scale study on the entire dataset showed that our method outperformed the baseline methods, which are based on the exact match between the aspect signature and sentences. Our method is able to annotate 61% more sentences than the baseline method with a precision of 0.8. The results also showed that there is a wide difference between the number of annotated sentences per movie category. In the worst case scenario we may only annotate eight sentences for a given movie-aspect pair.

A novel method to generate personalized aspect-based summaries The third contribution of this dissertation is a new method to generate personalized aspect based summaries of reviews. We formulate this problem as an optimization problem that balances two main objectives: to maximize the coverage of aspects the user is interested in and to maximize the representativeness of the opinions in the summary. The resulting formulation has the primary objective to maximize coverage of aspects in the user profile with the secondary objective to maximize representativeness. This optimization has constraints that ensure compactness and non-redundancy of the generated summary. We also propose a ranking function that computes the representativeness of a sentence for a particular aspect. The novelty of this function resides in using word embeddings to find groups of sentences with similar opinions about aspects. This function then rewards sentences that belong to popular groups. We denote the summary generated by our method as the mixed summary.

A set of user studies to evaluate user preferences toward personalized aspect-based summaries Our first user study shows that the majority of users agree with the groups of similar sentences we select. In addition, we found that not all users agree with the level of details in the summary. In fact we found that we have two categories of users: users who prefer detailed sentences in the summaries and others who prefer more concise and to the point sentences. Our second user study shows that users prefer the mixed summaries we propose in this dissertation over the fully-personalized summaries that contain only sentences that discuss the aspects the user is interested. We found that users complain most about the redundancy in he fully-personalized summaries, as most of the summary only discuss the few aspects the users are interested in. Our third user study shows that mixed summaries outperform the state-of-the-art LexRank method, which generates generic summaries. Users prefer the mixed summaries because they are not redundant and because they cover multiple aspects. This user study also points out to a number of interesting avenues for future research, which we discuss in the next section.

7.2 Future Work

The results of this dissertation point to a number of open problems to generate personalized, aspect-based summaries. The tools and techniques proposed in this dissertation are just an initial step towards building a platform that allows a complete personalized experience for users. We identify a number of research directions:

Summary visualization The summaries we generate can contain both positive and negative opinions about the same aspects. The results of our user study show that a

number of users dislike having contradictory opinions in the summary. This problem can be easily solved by better presenting the summaries to users. For example, we can first separate the positive from the negative opinions using different colors, and we can use a font size that is proportional to the representativeness of each sentence. This can help users understand that reviewers expressed contradicting opinion but with different proportions. Another possibility would be to generate two summaries for each movie: one with positive and the other with negative opinions. Beyond this issue of how to present different opinions, the length of the summaries is another important property. Further research is needed to evaluate how to determine the length given the device screen size and user preferences.

Improvement of the summary quality Our study shows that workers complain about the sentences readability. Workers dislike to have ill-formed sentences in the summary. This issue can be avoided by filtering out ill-formed sentences before running our optimization algorithm. Some workers dislike to see a summary as a list of disjoined sentences. To solve this problem, we can use an abstractive method to form a coherent text from the set of extracted sentences. The user study also shows that our grouping method suffered from three issues. The first issue is that our method fails to deal with the negation of opinions in the sentences. To solve this problem we need to use lexical rules to detect the negation in sentences, then to use a lexical database to associate the opinion negation with its antonym. The second issue is that our method does not consider the superlative adverbs used to intensify opinions in sentences. We can improve our method by grouping sentences using downtoners (i.e. slightly, barely, somewhat) and intensifiers (i.e. very, extremely, really) separately. The third issue is that our method considers that two sentences belong to the same group only if they contain similar opinions about the same aspect. Our user study showed that users find that two sentences are different when they contain a different information even when they contain the same opinion about the same aspect. One solution to this problem is to consider two sentences similar when they contain the same opinion about an aspect and carry overall the same information.

Multi-feature personalization Our current system only personalizes the summaries in terms of aspects. The user studies we conducted point out to a number of features that depend on the user preferences. First the level of detail in the summary. One of the findings of our user study is that some users prefer short and concise sentences in the summaries while others prefer more detailed and explanatory summaries. The summary format is another feature that we might add to the user profile. Our results show that some users prefer to have a list, which they consider easier to digest while others prefer to see a coherent paragraph. Our user study also shows that many users like the summaries we generate because they contain positive opinions about the aspects. We believe that in our system we should let the user decide if he/she wants a summary about the positive/negative opinions or both. The level of expertise may be another important feature as experts in movie critic may want to see more technical comments.

Implicit user profile generation Our current system requires users to manually enter their preferences for aspects. One area for future research is to develop methods to automatically build the user profiles. Automation is even more crucial if we consider multi-feature profiles that may become too much for users to fill out explicitly. One might infer user profiles implicitly based user interactions with movie

reviewing websites, for example, analyzing the reviews the user has written, or analyzing the reviews the user finds useful. We can also analyze the genre of movies the user usually watch, this can be a good indicator of the aspects the user is interested in.

Summarization in real time Our current system summarizes the reviews offline due to the relatively long time to solve the optimization problem. For example, it takes 3 minutes to generate the summary of the "Die Hard" movie that contains 294 opinionated sentences. Offline summarization worked for our user studies, because we consider a limited number of movies with a fixed set of reviews and a relatively few aspects for the user profile. Pre-computing summaries in advance, however is unpractical in an operational system. First the set of user profiles is large because it increases exponentially with the number of aspects (if we consider that the number of aspects is N than the total number of user profiles is 2^N). Second, users create content daily, which means that their profiles evolve in time and that the reviews per movie increase rapidly. In order to make fast responses to different user profiles, our method to generate summaries should run in real time.

Application of summarization method to other domains Although the methods developed in this disseration were applied only to movie reviews, they should be applicable to other domains where online reviews are prevelant (i.e. restaurants, hotels) or any set of opinionated text (i.e. weblogs). Further evaluation is needed to confirm this claim. In particular, two parts of our method need specific domain knowledge as input. First, the aspect signature extraction defined in Chapter 4 requires a set of aspect seed terms and fine tuning the algorithm parameters. Second, we manually correct the polarities of the opinion words before clustering them.

Appendix A

Appendix

A.1 Aspect signature results

A.1.1 Aspect noun signature

Algorithm parameters $\alpha = 1$, $d = 0.15$

TABLE A.1: Algorithm parameters $\alpha = 1$, $d = 0.15$

Aspect seed term	**Aspect noun signature**
fantasy	imaginings,wish-fulfillment, steampunk, dreamworld, surrealism, cyberpunk, fairy-tale, fable, futurism, wish-fulfillment, fiction, escapism, fairytale, fairy-tale, daydream, make-believe
science-fiction	sci-fi, science-fiction, scifi, sf
sound-effects	electric-guitar, synths, flashing-light, orchestration, philip-glass, soundscape, techno-music, synthesizer, sound-design, percussion, woodwind, synth, vangelis, accompaniment, sound-mixing, screeching, morricone, ambient, miklos, hans-zimmer, james-horner, synthesiser, howard-shore
biography	memoir, biopic, bio-pic, bio, autobiography
story	plot, tale, storyline
score	background-score, music-score, soundtrack, musical-score
script	screenplay
crime	felony, police-corruption, criminal-activity, murder, homicide, heinous-crime, mass-murder
romance	courtship, bromance, friendship, budding-romance, love-affair, unrequited-love, drama, flirtation, relationship, melodrama, lovey-dovey, sexual-tension
cinematography	camera-work, art-direction, production-design, photography, black-and-white-cinematography
animation	stop-motion-animation, stop-motion, computer-animation, claymation, animated-feature, cgi, pixar, hand-drawn-animation, hand-drawn
cast	nuanced-performance, portrayal, supporting-role, powerhouse-performance, perf, screen-presence, performace, role, stellar-cast, supporting-cast
comedy	screwball-comedy, humor, slapstick, romantic-comedy, humour, comedy-drama, satire, slapstick-comedy, raunchy-comedy, farce
visual-effect	visuals, sfx, special-fx, special-effect, f-x, fx, effect
horror	horror-flick, slasher, gothic-horror
makeup	make-up

Continued on next page

Table A.1 – *Continued from previous page*

Aspect seed term	**Aspect noun signature**
director	writer-director, filmmaker, helmer, writer-director, first-time-director
adventure	derring-do, joyride, roller-coaster-ride, spectacle, saga, caper, misadventure, treasure-hunt, voyage, epic, thrill-ride, swashbuckler, action-drama, adventurer, yarn, escapade, exotic-locale, saturday-matinee, odyssey, space-opera, extravaganza, journey, romp
news	announcement, news-report, bulletin, press, broadcaster, tabloid, cnn, newspaper, headline, newscaster, broadcasting, newscast, broadcast, report, coverage, front-page, news-anchor
thriller	suspense-thriller, crime-drama, psychological-thriller, political-thriller, crime-thriller, actioner, espionage-thriller, spy-thriller, supernatural-thriller, chiller
war	iraq-war, war, warfare, wartime, vietnam-war, civil-war
action	slam-bang, adrenaline, shoot'em-up, non-stop-action, high-octane, action-horror, explosion, actioneer, car-chase, shoot-em-up, all-action, action-thriller, suspense, swordplay, gunplay, action-film, gun-play
documentary	pseudo-documentary, docu-drama, documentary-style, mockumentary, docudrama, errol-morris, history-channel
musical	george-gershwin, astaire-rogers, broadway, cabaret, toe-tapping, cole-porter, song-and-dance, irving-berlin, jazz, busby-berkeley, songwriting, beatles, musical-number, al-jolson, bollywood, operetta, opera, gershwin, mgm, hammerstein, mikado, busby, classical-music, ragtime, cole-porter, berkeley, music, lyricist, ballad, bob-fosse, revue, rodgers, evita, bizet, song, dance-number, songbook, tune, easter-parade, broadway-melody
western	spaghetti-westerns

TABLE A.4: Positive adjectives clusters

Cluster number	**Positive adjectives**
0	[holy, spiritual, divine]
1	[concise, informative]
2	[innovative, visionary, unique, creative, bold, revolutionary, fresh, daring, ingenious, imaginative, inventive]
3	[veritable, genuine, pure, gleeful]
4	[meaningful, constructive, substantive, deep, sustainable, comprehensive, long-lasting, profound]
5	[insightful, adventuresome, intellectual, ambitious, erudite, eloquent, clever, sophisticated, intuitive, intelligent, cerebral, brainy, smart, introspective]
6	[patriotic, truthful, sincere, poetic, heartfelt, reverent, lyrical, respectful]
7	[uplifting, inspirational, impassioned, inspiring, audible]
8	[devout, avid, ardent, fervent]
9	[suspenseful, action-packed, exciting, thrilling, well-made, exhilarating]
10	[beloved, memorable, illustrious, classic, unforgettable, silent, famous, influential, iconic, legendary, immortal]
11	[undisputed, unbound, undisputable, unrivaled, unparalleled, irreplaceable, unmatched, unbreakable, unbeatable, hands-down]

Continued on next page

Table A.4 – *Continued from previous page*

Cluster number	Positive adjectives
12	[unabashed, abundant, boundless, everlasting, unfettered, irresistible, inestimable, unconditional, unquestionable, unequivocal, undeniable]
13	[comic, amusing, comical, humorous, comedic, humourous, funny, hilarious, lighthearted]
14	[desirable, tempting, fashionable, fond, complimentary, flattering, keen]
15	[funky, speedy, snappy, witty, lively, sparkling, brisk, sharp, zippy, sprightly, energetic, fast-paced, nimble, razor-sharp]
16	[effective, subtle, balanced, efficacious, complementary, timely, powerful, salutary, poignant]
17	[flashy, distinct, theatrical, stylized, judicious, stylistic, artistic, distinctive, stylish]
18	[world-famous, well-received, prolific, well-known, adulatory, popular, well-regarded, well-established, better-known, prominent]
19	[advantageous, auspicious, engrossing, innocuous, uncomplicated, eventful, intriguing, accessible, appropriate, agreeable, interesting, unconventional, unpredictable, instructive]
20	[dynamic, brotherly]
21	[funniest, freshest, greatest, pre-eminent, best-known, favorite, fav, finer, brightest, strongest, finest]
22	[quieter, sharper, lighter, softer, soft, warmer]
23	[cheapest, sharpest, smoothest, easiest, cleanest, nicest, coolest, smartest, fastest, neatest, toughest, luckiest, hottest]
24	[indulgent, congratulatory, trendy, artsy]
25	[positive, moderate, favorable, charitable, fair, merciful, generous, valid]
26	[gentle, soulful, humane, delicate, passionate, thoughtful, conciliatory, compassionate, sensitive, wise]
27	[amazing, remarkable, impressive, striking, enchanting, extraordinary, prodigious, awe-inspiring, astonishing, incredible, astounding]
28	[attune, intelligible, fluent, rational, lucid, vouch, articulate]
29	[suitable, catchy, easy, neat, heartening, welcome, nice, harmless, pleasant, painless, gratifying, lightweight]
30	[regal, magnetic, heavenly, angelic, rapturous, fiery, swashbuckling, ferocious, multi-purpose, courtly]
31	[hypnotic, dexterous, fluid, profuse, steady, spontaneous, purposeful, graceful, seamless, well-managed, rapid, crisp]
32	[top-quality, first-class, first-rate, pro, topnotch, strong, sturdy, top-notch, stellar]
33	[plentiful, monumental, stupendous, spectacular, resounding, nifty, state-of-the-art]
34	[fabulous, splendid, delicious, priceless, marvellous, wonderful, marvelous, delightful, felicitous]
35	[honorable, benevolent, ideal, unassailable, godlike, infallible, invincible, indestructible, heroic, altruistic, invulnerable]
36	[majestic, breathtaking, peerless, matchless, sublime, sensational, dazzling, exquisite, timeless, masterful, grand, magnificent, wondrous, glorious, wonderous, stunning, triumphant]
37	[firmer, richer, thinner, tougher, nicer, clearer, quicker, stronger, simpler]
38	[decent, great, good, solid]
39	[verifiable]

Continued on next page

Table A.4 – *Continued from previous page*

Cluster number	Positive adjectives
40	[consistent, reasonable, simplest, coherent, faithful, clear-cut, simplified, workable, satisfactory, straightforward, foolproof, clear, readable, transparent]
41	[preferable, convenient, invaluable, beneficial, fortuitous, cost-effective, valuable, important, prudent, selective, prompt, indebted, useful, handy, helpful]
42	[illuminating, intimate, romantic]
43	[pleasurable, tolerable, viewable, worth-while, worthwhile, enjoyable, worthy]
44	[well-balanced, compelling, fascinating, formidable, well-rounded, credible, well-written]
45	[desirous, considerate, well-educated, gracious, humble, trustworthy, polite, protective, loving, compliant, hard-working, cooperative, courteous, sociable, well-mannered, conscientious, friendly, industrious, trusting, solicitous, diligent, well-behaved, supportive]
46	[guiltless]
47	[impartial, diplomatic, objective, ethical, unbiased]
48	[oscar-nominated, deserving, outperformed, oscar-worthy]
49	[unforeseen, propitious, momentous, magical, miraculous, historic, rare]
50	[precise, adroit, immaculate, deft, intricate, skillful, meticulous, economical]
51	[individualized]
52	[responsive, amenable, enthral, proactive, attentive, knowledgeable, receptive, effusive, rapt, enthusiastic, appreciative, well-informed]
53	[courageous, valiant, resilient, undaunted, victorious, loyal, well-intentioned, principled, lionhearted, steadfast, fearless, noble, dauntless, decisive, unselfish, tenacious, resolute, magnanimous, virtuous, gallant, chivalrous, saintly]
54	[orderly, inviolable, exalted, unrestricted, equitable, eminent, amicable, autonomous, enterprising, exclusive, enviable, punctual]
55	[unaffected, idolized, unfazed, enthralled, undamaged, motivated, dumbfounded, reachable, impressed, stunned, unencumbered, unseen]
56	[thrifty, ebullient, authoritative, ingenuous, bullish, amiable, effectual, jovial, affable, congenial, effortless, easygoing, all-around, affectionate, genial, suave, exuberant, no-nonsense]
57	[achievable, admirable, meritorious, praiseworthy, noteworthy, laudable, significant, appreciable, commendable, superior]
58	[fragrant, lovely, swank, delectable, gorgeous, beautiful, sensual, alluring, beauteous, radiant, glamorous, luminous, vivacious, supple]
59	[feisty, gutsy, super, comely, cool, chic, cute, hot, good-looking, sporty, fast-growing, sexy, fine-looking, classy, handsome, attractive]
60	[understandable, logical, correct, affirmative, feasible, accurate]
61	[paramount, sufficient, proper]
62	[sleek, elegant, colourful, colorful, opulent, luxuriant, stately, lustrous, vivid, sumptuous, contrasty, evocative, lavish, plush, panoramic, expansive, fancy, eye-catching, luxurious, picturesque, bountiful, scenic, snazzy, resplendent, spacious, lush, vibrant]
63	[festive, bright, jubilant, euphoric, blithe, warm, joyous, lighthearted, playful, cheerful, cheery, adventurous, rosy, joyful, upbeat, high-spirited, carefree, warm-hearted, peppy, blissful, sweet, healthy, wholesome, youthful, chaste, bittersweet, buoyant, hopeful, optimistic]
64	[fresher, fancier, luckier, cleaner, cheaper, happier, brighter]

Continued on next page

Table A.4 – *Continued from previous page*

Cluster number	Positive adjectives
65	[inexpensive, usable, advanced, low-cost, extra, optimal, affordable, portable, available, unlimited, precious, ample]
66	[golden, prestigious, fertile, hallowed, detachable]
67	[satisfied, delighted, thankful, happy, overjoyed, grateful, pleased, honest, refreshed, safe, elated, glad, ready, eager, excited, fulfilled, willing, ecstatic]
68	[idyllic, harmonious, quiet, peaceful, tranquil, instrumental, peaceable, restful, serene, hospitable]
69	[free, hooray, fortunate, lucky]
70	[spotless, brand-new, comfy, clean, cozy, shiny, retractable, stainless]
71	[realistic, authentic, fantastical, unreal, dreamlike, surreal]
72	[bonny, jolly, hardy, merry]
73	[organized, non-violent, righteous, self-sufficient, civilized, sane, permissible, progressive, law-abiding, vigilant, upheld, blameless, lawful]
74	[envious, proud]
75	[modern, quaint, time-honored, halcyon, futuristic, nostalgic]
76	[excellent, adequate, exemplary, awesome, outstanding, exceptional, fantastic, flawless, phenomenal, brilliant, irreproachable, terrific, dead-on, faultless, perfect, superb, impeccable]
77	[robust, confident, flexible, agile, gifted, efficient, talented, capable, skilled, resourceful, replaceable, adaptable, savvy, competitive, proficient, adaptive, durable, versatile]
78	[prosperous, wealthy, palatial, swanky, well-connected, rightful, gainful, frugal, affluent, roomy, rich]
79	[successful, modest, productive, respectable, lucrative, professional, fruitful, promising, reputable, competent, reliable, consummate, stable, seasoned]
80	[manageable, compact, cohesive, tidy, risk-free, lean]
81	[lovable, relatable, adorable, believable, sympathetic, charismatic, dependable, likeable, charming, believeable, likable, endearing]
82	[cushy, well-bred, posh, privileged, pampered]
83	[fit, comfortable, compatible]

TABLE A.5: Negative adjectives clusters

Cluster number	Negative adjectives
0	[dark, uglier, darker, narrower, darkest]
1	[poorest, slowest, oddest, strangest, stupidest, worst, spookiest]
2	[objectionable, inappropriate, distasteful, unacceptable, demeaning, disgraceful, off-putting, degrading, indefensible, blatant, derogatory, blasphemous, offensive]
3	[divergent, audacious, fanciful, deceptive, tenuous, inextricable, provocative, complex, paradoxical, antithetical, cautionary, satirical, rhetorical, perplexing, subversive, nebulous, contradictory]
4	[controversial, discriminatory, extremist, disputable, anti-white, radical, conservative, prejudicial, inflammatory, left-leaning, hard-line, anti-american, divisive, reactionary, vehement]

Continued on next page

Table A.5 – *Continued from previous page*

Cluster number	**Negative adjectives**
5	[spooky, horrifying, bloody, disconcerting, creepy, unpleasant, grisly, torturous, excruciating, shocking, gruesome, distressing, frightening, unsettling, startling, frightful, infuriating, uncomfortable, scary, disturbing, maddening, painful, agonizing, horrific]
6	[loud, overbearing, boisterous, dissonant, shrill, discordant, squeaky, nagging, obtrusive, noisy, abrasive, strident, incessant, intrusive]
7	[caustic, sober, incomparable, sly, sarcastic, facetious, monotone, acerbic, mordant, sardonic, laconic, glib]
8	[foul, depraved, ghastly, repulsive, revolting, rotten, twisted, nasty, freakish, ugly, heinous, sinful, disgusting, filthy, sickening, perverse, vile, hideous, grotesque, unspeakable]
9	[peculiar, unorthodox, irregular, emphatic, anomalous, ironical, ominous, ironic, uneasy, equivocal, oblique, obscure, urgent, odd, ambiguous, invasive, illusory, unusual]
10	[enraged, mortified, dismayed, outraged, aghast]
11	[sporadic, fleeting, pique, faint]
12	[unfulfilled, devoid, soulless, bereft, hollow, false]
13	[mischievous, fiendish, shadowy, devious, beastly, wicked, nefarious, crafty, maniacal, menacing, devilish, demonic, mysterious, malevolent, insidious, villianous, malignant, villian, diabolical, sneaky, malicious, murderous, dastardly, vengeful, venomous, diabolic, fearsome, sinister, treacherous, monstrous, villainous]
14	[uncontrolled, destructive, undermined, ruinous, conspicuous, indiscriminate, damnable, wanton, scathing, rampant, war-like, pernicious, corrosive, pervasive, harmful, rife, virulent, flagrant, noxious]
15	[unyielding, feverish, blinding, oppressive, implacable, unrelenting, acrid, uncompromising, intense, relentless, overwhelming, oppressiveness]
16	[cruelest]
17	[negative, erroneous, fallacious, distorted, counterproductive, bothersome, simplistic, pedantic, fatuous, one-sided, unreasonable, superficial, obtuse, self-defeating, disingenuous, spurious, trivial, untrue, incorrect, inaccurate]
18	[stereotypical, stereotyped, cliched, queer]
19	[somber, morbid, austere, solemn, pessimistic, macabre, nightmarish, mournful, irreversible, impersonal, disquieting, fatalistic, despairing, downcast, infernal]
20	[die-hard, watered-down, bloated, noisier]
21	[incomplete, abrupt, unsatisfying, niggle, unsatisfactory, disappointing, dissatisfying, puzzling, unfinished, baffling]
22	[superbad, naughty, smutty, obscene, lewd, gross, tasteless, profane, rude, childish, uproarious, outrageous, coarse, offensiveness, racy, crass, vulgar, splatter, dirty]
23	[lugubrious, mushy, overblown, farcical, melodramatic, drippy, soapy, mawkish, full-blown, bombastic, heavy-handed, sugar-coated]
24	[unfounded, unwarranted, unjustifiable, undue, groundless, unsupported, unjustified, baseless]
25	[byzantine, dense, tangled, knotted, unwieldy]
26	[idiotic, flat-out, stupid, moronic, senseless, mindless, idiot, pathetic, dumb, retarded, inane, shameless, unfunny, asinine]

Continued on next page

Table A.5 – *Continued from previous page*

Cluster number	Negative adjectives
27	[deplorable, horrendous, pitiful, abysmal, atrocious, misbegotten, appalling, woeful, dreadful, wretched, dire, god-awful, horrid, inexcusable, abominable, lamentable]
28	[low-budget, gritty, grim, stark, taut, bleak, grainy, washed-out, moody, gloomy]
29	[unfortunate, shameful, detrimental, problematic, critical, regrettable, galling, disheartening, egregious, belated, dismaying, debatable]
30	[wasteful, clogged, meaningless]
31	[untruthful, immoral, slanderous, condescending, dismissive, insensitive, dishonest, irrational, loath, thoughtless, hurtful, sanctimonious, unkind, unethical, derisive, ignorant, disrespectful, presumptuous, shortsighted, insincere, hypocritical, contemptuous, hateful]
32	[inactive, unprofitable, unavailable, ineligible]
33	[rocky, fucking, furious, craven]
34	[notorious, fainthearted, infamous]
35	[unseemly, ignominious, imprudent, indelicate, involuntary, indecent, injurious, inordinate, unwelcome, untenable, adverse, intractable, irreconcilable, unequal, improper, ignoble, allergic, untimely, unfavorable, insupportable, adversarial]
36	[manic, weird, batty, crazy, bizarre, hysterical, goofy, eccentric, delirious, strange, kooky]
37	[nonsensical, far-fetched, fake, unbelievable, cheesy, ridiculous, incredulous, preposterous, illogical, tacky, laughable, unrealistic, ludicrous, bogus, implausible, improbable, cartoonish, absurd, farfetched]
38	[adamant, ambivalent, questionable, carp, ulterior, marginal, dubious, doubtful, vague]
39	[heartbreaking, tragic, sad, sorrowful]
40	[bitter, brutal, inhuman, unjust, pitiless, harsh, violent, merciless, inhumane, vicious, unforgiving, humiliating, cruel, remorseless]
41	[hazy, discontinuous, desultory, hasty, haphazard, sloppy, messy, disorganized, fragmented, imprecise, incoherent, blurry, choppy]
42	[malign, traitorous, fanatical, treasonous, plebeian, invidious, hawkish, autocratic, despotic, heretic, heretical, rival, retaliatory, anarchistic, dishonorable, unconstitutional, zealous, totalitarian, dictatorial, conspiratorial, oppress, resurgent, fascist, tyrannical, partisan, draconian]
43	[unreadable, unworthy, inaudible, painfull, inconceivable, impossible, intolerable, unusable, indiscernible, nonexistent, unbearable, inferior, unworkable, indistinguishable, incomprehensible, worthless, unwatchable, unintelligible]
44	[lousy, bad, horrible, awful, sucky, shoddy, lame, terrible, lazy, crappy, poor, crummy]
45	[toxic, contagious, infectious]
46	[cannibal, barbarous, puny, damned, uncivil, primitive, superstitious, impure, barbaric, bestial, uncivilized, heathen, predatory, rabid, savage, warlike, bloodthirsty]
47	[debauch]
48	[complacent, cynical, unsophisticated, fickle, uninformed, unaccustomed, indiscriminating, spoon-fed, susceptible, jaundiced, credulous, jaded]
49	[unimaginative, anemic, insubstantial, shallow, dreary, uncreative, ill-conceived, lifeless, aimless, uninspired, bland, spiritless, stodgy, uninteresting, banal, listless, pretentious, insipid]

Continued on next page

Table A.5 – *Continued from previous page*

Cluster number	Negative adjectives
50	[fat, dead, poisonous, famished, fatty, infected, fried, thicker, cancerous]
51	[counter-productive]
52	[spotty, mediocre, dismal, lackluster, substandard, second-tier, ho-hum, half-hearted, moribund, limp, tepid, lukewarm]
53	[unraveled, immaterial, insignificant, inconsequential, unimportant, sketchy, undefined, unresolved, unexplained, underdeveloped, ill-defined, moot, unclear]
54	[inconsolable, irretrievable, irreparable]
55	[forgetful, over-hyped, uneventful, unremarkable, unoriginal]
56	[neglected, deformed, dissed, belittled, castrated, disadvantaged, scarred, abused, handicapped, disabled, tortured]
57	[boggle, indeterminable, unsound, untested]
58	[irksome, tired, annoying, tiresome, aggravating, irritating, wearisome]
59	[over-priced, arcane, outmoded, impractical, antiquated, obsolete, expensive, archaic, passe, extravagant, unsustainable]
60	[undependable]
61	[disobedient, brutish, heartless, capricious, callous, smug, indecisive, passive, submissive, disloyal, deceitful, egotistical, careless, aloof, impudent, self-interested, self-serving, disdainful, distrustful, egocentric, manipulative, prideful, revengeful, mendacious, argumentative, stubborn, spiteful, arrogant, opinionated, belligerent, vindictive, avaricious, cowardly, scornful, greedy, selfish, mistrustful, uncaring]
62	[slow-moving, monotonous, sluggish, protracted, lethargic, stagnant, repetitive, gimmicky, tedious, dull, overused, intermittent, stale, cumbersome, lengthy, redundant, static]
63	[infidel, iniquitous, hell-bent, inglorious]
64	[gawky, lovelorn, pompous, fussy, inapt, lier, stingy, bashful, ill-used, flakey, feckless, fusty, lecherous, gauche, needy, frigid, flighty, harpy, bitchy, fretful, timid, two-faced, flaky, curt, sullen, irrepressible, snobbish, crabby, lech, choleric, heartbreaker, absent-minded, boastful, fastidious, covetous, imperious, haughty, perfidious, quarrelsome, lascivious, stuffy]
65	[pokey, sickly, itchy, saggy, plasticky, haggard, ragged, greasy, dim, smudged, lumpy, sticky, chunky, lanky, glum, bulky, forlorn, unclean, wrinkled, smelly, sweaty, tattered, fuzzy, woebegone, skinny]
66	[clunky, perfunctory, underpowered, over-acted, unconvincing, unnatural, awkward, clumsy, inconsistent, strained, unauthentic, lackadaisical, stiff]
67	[regressive]
68	[fraught, inexorable, cataclysmic, tumultuous, inevitable, perilous, unthinkable, fateful, turbulent, apocalyptic, catastrophic, calamitous, unimaginable, insurmountable, inescapable]
69	[unnecessary, arbitrary, miscellaneous, incongruous, dispensable, irrelevant, superfluous, extraneous, useless, excessive, inessential, unneeded, pointless]
70	[grumpier, rusty, stern, fluster, weary, infirm, doddering, pugnacious, rough, grouchy, irascible, grumpy, finicky, grouch, madden, tetchy, miserly, cocky, gruff, cranky, wily, glower]
71	[disapproving, panicky, evasive, mispronounced, reluctant, reticent, testy, skittish, distressed, nervous, bemused, indignant, laid-off, exasperated, long-time]
72	[idle, overpriced]
73	[uneven, feeble, hampered, faltered, flimsy, weaker, weak, poorer]

Continued on next page

Table A.5 – *Continued from previous page*

Cluster number	Negative adjectives
74	[guilty, premeditated, punishable, culpable, wrongful, fraudulent, unlawful, punitive]
75	[time-consuming, costly, tortuous]
76	[congested, slogged]
77	[contentious, touchy, difficult, thorny, tricky]
78	[grievous, burdensome, drastic, debilitating, life-threatening, severe, fatal, ignominy, chronic, traumatic]
79	[repulse, dizzy, hysteric, giddy, liable, numb, sore, unsettle, sick]
80	[wrong, berserk, bonkers, sour, bankrupt, untouched, unnoticed, amiss]
81	[disastrous, overdue, acrimonious, ill-fated, bungle, ill-advised, disasterous, short-lived, unsuccessful, flagging, unconfirmed, accidental, unfaithful, indiscreet, low-rated, unachievable]
82	[overdo, feign, overact, mangle, lambast, dawdle, recant, mock, defame]
83	[officious, antagonistic, impotent, incompetent, uncooperative, opportunistic, overzealous, unproductive, inefficient, underpaid, untrustworthy, inimical, unreliable, ineffectual, embattled, ineffective, inept, obstinate, insubordinate, unhelpful, negligent, unskilled, unscrupulous, ill-tempered, inexperienced, unsupportive]
84	[hurt, mangled, hung, petrified, dislocated, invisible, paralyzed, defective, swollen, crushed]
85	[bumpy, unsteady, frenetic, jittery, scratchy, shaky, jerky, hectic, restless, slower, languid, frantic, jumpy, frenzied, sedentary, languorous, erratic, chaotic]
86	[litigious, coercive, stringent, intransigent, insular, authoritarian, repressive, second-class, inflexible, restrictive, rigid, strict, dogmatic]
87	[ordinary, imperfect, simple, mundane]
88	[uninsured, unemployed, illegitimate, unwanted, oversize, undocumented, unlicensed, irate, unproven, indeterminate, impoverished, indigent, abusive, errant, invalid, inoperable, aweful, unspecified, orphan, absentee, undetermined, accursed, incorrigible]
89	[degenerate, tawdry, salacious, lurid, scandalous, gaudy, hedonistic, decadent, hothouse, licentious, garish, sleazy]
90	[vexing, precarious, stressful, fractious, inopportune, unfriendly, troublesome, disruptive, unsafe, worrisome, dangerous, secretive, unsavory]
91	[despondent, broken-hearted, remorseful, resentful, grudging, distraught, dazed, dispirited, regretful]
92	[subservient, loathsome, disagreeable, despicable, deficient, irredeemable, pitiable, repugnant, reprehensible, contemptible, unlikable, detestable]
93	[vociferous, combative, impetuous, forceful, brazen, heedless, aggressive, brash, defiant, volatile, impulsive, confrontational, fierce, reckless]
94	[averse, nitpicking, unsure, puzzled, undecided, hesitant, flabbergasted, unprepared, perplexed, unmoved, floored, uncertain]
95	[unpopular, unqualified, unlucky, unwise, unlikely]
96	[immature, undesirable, indifferent, illiterate, inarticulate, uncouth, irresponsible, obese, impertinent, inconsiderate, slothful, impolite, unfit, unattractive, unresponsive, obnoxious, ungrateful, insufferable, irritable, insolent, ill-mannered, apathetic, inattentive, indolent, miserable, inconstant, oblivious]

Continued on next page

Table A.5 – *Continued from previous page*

Cluster number	**Negative adjectives**
97	[frozen, leaky, cloudy, dusty, cold, murky, stormy, desolate, pacific, steep, fetid, barren, humid, inclement, cloud, motionless, inhospitable, chilly, muddy, precipitous, sunken, jagged]
98	[lethal, unexpected, assault, explosive, tense, incendiary, deadly]
99	[faulty, unable, scarce, needless, insufficient, inadequate]
100	[immodest, skimpy, smallish, blotchy, top-heavy, lesser-known, shabby, high-priced]
101	[naive, foolhardy, hopeless, desperate, futile, foolish, gullible, misguided, vain, petty]
102	[meager, hefty, disproportionate, exorbitant, pricey, paltry, prohibitive, measly, scant]
103	[insecure, obsessive, compulsive, insatiable, anti-social, instable, abnormal, unreachable, unhealthy, obsessiveness, phobic, unstable, self-destructive, neurotic, illicit, immovable]
104	[fictional, unfamiliar, fictitious, little-known, imaginary]
105	[vagrant, truant, drop-out, unsuspecting, unruly, recalcitrant, runaway, delinquent, wayward, rebellious, lowly]
106	[envious, wary, jealous, suspicious, ashamed, afraid, fearful]
107	[lawless, wild, cutthroat]
108	[strangle, finagle, break, splitting, miss]
109	[menial, fruitless, thankless, strenuous, risky, arduous, hazardous, last-ditch, onerous]
110	[fragile, vulnerable, frail, brittle]
111	[disturbed, annoyed, dissatisfied, sceptical, irritated, angry, confused, disgusted, frustrated, perturbed, unnerved, bored, overwhelmed, disinterested, picky, anxious, apprehensive, unsettled, livid, scared, leery, skeptical]
112	[suicidal, incapable, deaf, unwilling, helpless, deluded, unobserved, senile, mad, delusional, blind, lunatic, paranoid, insane, powerless]
113	[ruthless, disreputable, corrupt, felonious, lawbreaker, crooked, rogue, shady, fugitive, miscreant, illegal, criminal, hooligan]
114	[incompatible, misfit, maladjusted, motley, unhappy, dysfunctional, discontented, disordered, disaffected]
115	[lonesome, disorderly, tardy, drunken, hapless, tramp, drunk, lone]
116	[drab, flat]
117	[decayed, cash-strapped, lonely, ramshackle, destitute, musty, dilapidated, cramped, jobless, run-down, seedy, decrepit]

TABLE A.2: Table that summarizes the notations used in this dissertation

Notation	Definition
$M = \{m_1, ..., m_l\}$	Set of movies in the corpus
l	Number of movies in the corpus
$R_m = \{r_{m1}, ..., r_{mq}\}$	Set of reviews per movie
$S_r = \{s_{r1}, ..., s_{rp}\}$	Set of sentences per review
V	Vocabulary
h	Vocabulary size
W	Word embedding matrix
$\vec{v}$	Vector representation of a word w
dim	Word vector representation dimension
c	Context word
$freq_w$	Frequency of the word w
$A = \{a_1, \ldots, a_n\}$	List of aspects
n	Number of aspects
k	Aspect signature length
$ASN(a_i)$	List of noun terms in the signature of an aspect a_i
α	Degree of overlap
d	Similarity distance
ASM	Aspect signature matrix where each row i contains the signature of nouns of the aspect a_i
$A(s_j)$	The set of aspects discussed in sentence s_j
$SN(s_j)$	Noun terms in the sentence s_j
θ	Similarity threshold for annotating sentences
AN_u	Sentence annotation by users
AN_{alg}	Sentence annotation by our algorithm
O_{a_i}	List of opinionated sentences for the aspect a_i
$O_m = \bigcup_{i=1}^{n} O_{a_i}$	List of opinionated sentences for the movie m
o	Opinionated sentence in O_m
$U = \{ u_{a1}, ..., u_{an}\}$	A boolean feature vector that represents the user profile. Each feature represents an aspect
Z	The personalized aspect-based summary we generate s.t. $Z \subset O_m$
x_a	Boolean variable that is equal to 0 if a sentence discussing the aspect a is covered in Z and 1 otherwise
$f(o, a)$	Representativeness scoring function of the sentence o for the aspect a
sim_a	Similarity between two opinion terms with respect to the opinion about a
δ_l	Summary size i.e. number of words
δ_{sim}	Similarity threshold between two opinion terms in the summary Z
C_p	Set of positive opinion clusters
C_n	Set of negative opinion clusters
$C = C_p \cup C_n = \{c_1, ..., c_t\}$	Set of all opinion clusters
$O_a = \{G_{a,c_1}, ..., G_{a,c_t}\}$	Set of opinionated sentences for aspect a_i where each group of sentences $G_{a,c}$ contains sentences with opinion terms from the same cluster c
$x_{G_a^o}$	Boolean variable that is equal to 0 if a sentence from a group of sentences G_a^o is covered in Z and 1 otherwise

TABLE A.3: List of manually added positive and negative opinion adjectives

Positive adjectives		**Negative adjectives**
cerebral		
lightweight	introspective	
rare	unconventional	
informative	magnetic	
sympathetic	colourful	unlikable
awe-inspiring	neo	unimaginative
lighthearted	relatable	low-budget
exclusive	theatrical	ordinary
comic	bold	simple
action-packed	intellectual	underdeveloped
fantastical	funniest	uninspired
good-looking	deep	dysfunctional
organized	comedic	uninteresting
pro	dreamlike	unoriginal
unique	distinct	unfunny
iconic	suspenseful	superbad
fit	oscar-nominated	darkest
well-written	oscar-worthy	infectious
swashbuckling	hypnotic	tasteless
objective	compelling	pacific
professional	moderate	stereotyped
fulfilled	undeniable	unsatisfying
civilized	fluid	flat
ferocious	valid	overused
stylistic	timeless	soulless
indestructible	artsy	monotone
sensual	competent	off-putting
immortal	historic	unremarkable
subtle	unseen	insipid
no-nonsense	extra	nonsensical
comical	likeable	
unforeseen	artistic	
funky	bittersweet	
funnily		

Bibliography

[1] Ayoub Bagheri, Mohamad Saraee, and Franciska de Jong. "An unsupervised aspect detection model for sentiment analysis of reviews". In: *International Conference on Application of Natural Language to Information Systems*. Springer. 2013, pp. 140–151.

[2] Wouter Bancken, Daniele Alfarone, and Jesse Davis. "Automatically detecting and rating product aspects from textual customer reviews". In: *Proceedings of the 1st international workshop on interactions between data mining and natural language processing at ECML/PKDD*. 2014, pp. 1–16.

[3] Evangelos Banos et al. "PersoNews: a personalized news reader enhanced by machine learning and semantic filtering". In: *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer. 2006, pp. 975–982.

[4] Regina Barzilay, Noemie Elhadad, and Kathleen R McKeown. "Sentence ordering in multidocument summarization". In: *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics. 2001, pp. 1–7.

[5] Shlomo Berkovsky, Timothy Baldwin, and Ingrid Zukerman. "Aspect-based personalized text summarization". In: *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems*. Springer. 2008, pp. 267–270.

[6] Purnima Bholowalia and Arvind Kumar. "EBK-means: A clustering technique based on elbow method and k-means in WSN". In: *International Journal of Computer Applications* 105.9 (2014).

[7] Steven Bird. "NLTK: the natural language toolkit". In: *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics. 2006, pp. 69–72.

[8] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[9] Sasha Blair-Goldensohn et al. "Building a sentiment summarizer for local service reviews". In: *WWW workshop on NLP in the information explosion era*. Vol. 14. 2008, pp. 339–348.

[10] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: vol. 3. JMLR. org, 2003.

[11] SRK Branavan et al. "Learning document-level semantic properties from free-text annotations". In: *Journal of Artificial Intelligence Research* 34 (2009), pp. 569–603.

[12] Marco Campana and Anastasios Tombros. "Incremental personalised summarisation with novelty detection". In: *International Conference on Flexible Query Answering Systems*. Springer. 2009, pp. 641–652.

[13] Kenneth Church et al. “Parsing, word associations and typical predicate-argument relations”. In: *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics. 1989, pp. 75–81.

[14] Rudi L Cilibrasi and Paul MB Vitanyi. “The google similarity distance”. In: *IEEE Transactions on knowledge and data engineering* 19.3 (2007).

[15] Ronan Collobert et al. “Natural language processing (almost) from scratch”. In: *Journal of Machine Learning Research* 12.Aug (2011), pp. 2493–2537.

[16] ILOG Cplex. “7.0 Reference Manual”. In: *Accessed as HTML document, distributed with ILOG CPLEX* 7 (2000).

[17] Hal Daume III and Daniel Marcu. “Domain adaptation for statistical classifiers”. In: *Journal of artificial Intelligence research* 26 (2006), pp. 101–126.

[18] Scott Deerwester et al. “Indexing by latent semantic analysis”. In: *Journal of the American society for information science* 41.6 (1990), p. 391.

[19] Jacob Devlin et al. “Fast and Robust Neural Network Joint Models for Statistical Machine Translation.” In: *ACL (1)*. 2014, pp. 1370–1380.

[20] Qiming Diao et al. “Jointly modeling aspects, ratings and sentiments for movie recommendation (JMARS)”. In: *Proc. of ACM SIGKDD*. 2014.

[21] Alberto Díaz and Pablo Gervás. “User-model based personalized summarization”. In: *Information Processing & Management* 43.6 (2007), pp. 1715–1734.

[22] Alberto Díaz, Pablo Gervás, and Antonio García. “Evaluation of a system for personalized summarization of web contents”. In: *International Conference on User Modeling*. Springer. 2005, pp. 453–462.

[23] Harris Drucker et al. “Support vector regression machines”. In: *Advances in neural information processing systems*. 1997, pp. 155–161.

[24] Carl Eckart and Gale Young. “The approximation of one matrix by another of lower rank”. In: *Psychometrika* 1.3 (1936), pp. 211–218.

[25] Harold P Edmundson. “New methods in automatic extracting”. In: *Journal of the ACM (JACM)* 16.2 (1969), pp. 264–285.

[26] Günes Erkan and Dragomir R Radev. “LexRank: Graph-based lexical centrality as salience in text summarization”. In: *Journal of Artificial Intelligence Research* 22 (2004), pp. 457–479.

[27] Alona Fyshe et al. “A Compositional and Interpretable Semantic Space.” In: *HLT-NAACL*. 2015, pp. 32–41.

[28] Mahak Gambhir and Vishal Gupta. “Recent automatic text summarization techniques: a survey”. In: *Artificial Intelligence Review* 47.1 (2017), pp. 1–66.

[29] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. “Opinosis: A graph based approach to abstractive summarization of highly redundant opinions”. In: (2010).

[30] Jade Goldstein et al. “Multi-document summarization by sentence extraction”. In: *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization*. Association for Computational Linguistics. 2000, pp. 40–48.

[31] Vishal Gupta and Gurpreet Singh Lehal. “A survey of text summarization extractive techniques”. In: *Journal of emerging technologies in web intelligence* 2.3 (2010), pp. 258–268.

[32] Zellig S Harris. “Distributional structure”. In: *Word* 10.2-3 (1954), pp. 146–162.

[33] Leonhard Hennig and Sahin Albayrak. "Personalized multi-document summarization using n-gram topic model fusion". In: *Information Management SPIM 2010* (2010), p. 24.

[34] Thomas Hofmann. "Probabilistic latent semantic indexing". In: *ACM SIGIR Forum*. Vol. 51. 2. ACM. 2017, pp. 211–218.

[35] Eduard Hovy and Chin-Yew Lin. "Automated text summarization and the SUMMARIST system". In: *Proceedings of a workshop on held at Baltimore, Maryland: October 13-15, 1998*. Association for Computational Linguistics. 1998, pp. 197–214.

[36] Su Su Htay and Khin Thidar Lynn. "Extracting product features and opinion words using pattern knowledge in customer reviews". In: *The Scientific World Journal* 2013 (2013).

[37] Minqing Hu and Bing Liu. "Mining and summarizing customer reviews". In: *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2004.

[38] Minqing Hu and Bing Liu. "Mining opinion features in customer reviews". In: *AAAI*. Vol. 4. 2004, pp. 755–760.

[39] Minqing Hu and Bing Liu. "Opinion extraction and summarization on the web". In: *AAAI*. Vol. 7. 2006, pp. 1621–1624.

[40] Jing Jiang and ChengXiang Zhai. "Instance weighting for domain adaptation in NLP". In: *Proceedings of the 45th annual meeting of the association of computational linguistics*. 2007, pp. 264–271.

[41] Wei Jin, Hung Hay Ho, and Rohini K Srihari. "OpinionMiner: a novel machine learning system for web opinion mining and extraction". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2009, pp. 1195–1204.

[42] Hongyan Jing. "Using hidden Markov modeling to decompose human-written summaries". In: *Computational linguistics* 28.4 (2002), pp. 527–543.

[43] Nal Kalchbrenner and Phil Blunsom. "Recurrent Continuous Translation Models." In: *EMNLP*. Vol. 3. 39. 2013, p. 413.

[44] Hyun Duk Kim et al. *Comprehensive review of opinion summarization*. Tech. rep. 2011.

[45] Kevin Knight and Daniel Marcu. "Statistics-based summarization-step one: Sentence compression". In: *AAAI/IAAI* 2000 (2000), pp. 703–710.

[46] Nozomi Kobayashi, Kentaro Inui, and Yuji Matsumoto. "Extracting aspect-evaluation and aspect-of relations in opinion mining". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. 2007.

[47] Lingpeng Kong et al. "A dependency parser for tweets". In: (2014).

[48] Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen. "Opinion Extraction, Summarization and Tracking in News and Blog Corpora." In: *AAAI spring symposium: Computational approaches to analyzing weblogs*. 2006.

[49] Chandan Kumar, Prasad Pingali, and Vasudeva Varma. "Generating personalized summaries using publicly available web documents". In: *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 03*. IEEE Computer Society. 2008, pp. 103–106.

[50] Julian Kupiec, Jan Pedersen, and Francine Chen. "A trainable document summarizer". In: *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 1995, pp. 68–73.

[51] John Lafferty, Andrew McCallum, and Fernando CN Pereira. "Conditional random fields: Probabilistic models for segmenting and labeling sequence data". In: (2001).

[52] Omer Levy, Yoav Goldberg, and Ido Dagan. "Improving distributional similarity with lessons learned from word embeddings". In: *Transactions of the Association for Computational Linguistics* 3 (2015), pp. 211–225.

[53] Fangtao Li et al. "Structure-aware review mining and summarization". In: *Proceedings of the 23rd international conference on computational linguistics*. Association for Computational Linguistics. 2010, pp. 653–661.

[54] Bing Liu. "Sentiment analysis and opinion mining". In: *Synthesis lectures on human language technologies* 5.1 (2012), pp. 1–167.

[55] Bing Liu, Minqing Hu, and Junsheng Cheng. "Opinion observer: analyzing and comparing opinions on the web". In: *Proceedings of the 14th international conference on World Wide Web*. 2005.

[56] Bing Liu, Minqing Hu, and Junsheng Cheng. "Opinion observer: analyzing and comparing opinions on the web". In: *Proceedings of the 14th international conference on World Wide Web*. ACM. 2005, pp. 342–351.

[57] Chong Long, Jie Zhang, and Xiaoyan Zhut. "A review selection approach for accurate feature rating estimation". In: *Proceedings of the 23rd international conference on computational linguistics: Posters*. Association for Computational Linguistics. 2010, pp. 766–774.

[58] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. "Rated aspect summarization of short comments". In: *Proceedings of the 18th international conference on World wide web*. ACM. 2009, pp. 131–140.

[59] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. "Rated aspect summarization of short comments". In: *Proceedings of the 18th international conference on World wide web*. 2009.

[60] Hans Peter Luhn. "The automatic creation of literature abstracts". In: *IBM Journal of research and development* 2.2 (1958), pp. 159–165.

[61] Laurens van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov (2008), pp. 2579–2605.

[62] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[63] Inderjeet Mani and Eric Bloedorn. "Summarizing similarities and differences among related documents". In: *Information Retrieval* 1.1-2 (1999), pp. 35–67.

[64] Christopher D Manning, Hinrich Schütze, et al. *Foundations of statistical natural language processing*. Vol. 999. MIT Press, 1999.

[65] Daniel Marcu. "From discourse structures to text summaries". In: *Intelligent Scalable Text Summarization* (1997).

[66] Julian McAuley and Jure Leskovec. "Hidden factors and hidden topics: understanding rating dimensions with review text". In: *Proceedings of the 7th ACM conference on Recommender systems*. ACM. 2013, pp. 165–172.

[67] Qiaozhu Mei et al. "Topic sentiment mixture: modeling facets and opinions in weblogs". In: *Proceedings of the 16th international conference on World Wide Web*. ACM. 2007, pp. 171–180.

[68] Qiaozhu Mei et al. "Topic sentiment mixture: modeling facets and opinions in weblogs". In: *Proceedings of the 16th international conference on World Wide Web*. 2007.

[69] Rada Mihalcea and Paul Tarau. "A language independent algorithm for single and multiple document summarization". In: *Companion Volume to the Proceedings of Conference including Posters/Demos and tutorial abstracts*. 2005.

[70] Tomas Mikolov et al. "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems*. 2013.

[71] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[72] George A Miller. "WordNet: a lexical database for English". In: *Communications of the ACM* 38.11 (1995), pp. 39–41.

[73] Róbert Móro et al. "Personalized text summarization based on important terms identification". In: *Database and Expert Systems Applications (DEXA), 2012 23rd International Workshop on*. IEEE. 2012, pp. 131–135.

[74] Olfa Nasraoui. "World wide web personalization". In: *Encyclopedia of data warehousing and mining*. IGI Global, 2005, pp. 1235–1241.

[75] Ani Nenkova, Kathleen McKeown, et al. "Automatic summarization". In: *Foundations and Trends® in Information Retrieval* 5.2–3 (2011), pp. 103–233.

[76] Tadashi Nomoto and Yuji Matsumoto. "A new approach to unsupervised text summarization". In: *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2001, pp. 26–34.

[77] Lawrence Page et al. *The PageRank citation ranking: Bringing order to the web.* Tech. rep. Stanford InfoLab, 1999.

[78] Bo Pang and Lillian Lee. "Opinion mining and sentiment analysis". In: *Foundations and trends in information retrieval* (2008).

[79] Jaehui Park et al. "Web content summarization using social bookmarks: A new approach for social summarization". In: *Proceedings of the 10th ACM workshop on Web information and data management*. ACM. 2008, pp. 103–110.

[80] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global Vectors for Word Representation." In: *EMNLP*. Vol. 14. 2014, pp. 1532–1543.

[81] Ana-Maria Popescu and Orena Etzioni. "Extracting product features and opinions from reviews". In: *Natural language processing and text mining*. Springer, 2007, pp. 9–28.

[82] Ana-Maria Popescu and Orena Etzioni. "Extracting product features and opinions from reviews". In: *Natural language processing and text mining*. 2007.

[83] Guang Qiu et al. "Opinion word expansion and target extraction through double propagation". In: *Computational linguistics* 37.1 (2011), pp. 9–27.

[84] Lawrence R Rabiner. "A tutorial on hidden Markov models and selected applications in speech recognition". In: *Proceedings of the IEEE* 77.2 (1989), pp. 257–286.

[85] Dragomir R Radev, Sasha Blair-Goldensohn, and Zhu Zhang. "Experiments in single and multidocument summarization using MEAD". In: *First document understanding conference*. Citeseer. 2001, pp. 1–7.

[86] Dragomir R Radev and Kathleen R McKeown. "Generating natural language summaries from multiple on-line sources". In: *Computational Linguistics* 24.3 (1998), pp. 470–500.

[87] GJ Rath, A Resnick, and TR Savage. "The formation of abstracts by the selection of sentences. Part I. Sentence selection by men and machines". In: *American Documentation* 12.2 (1961), pp. 139–141.

[88] Horacio Saggion and Thierry Poibeau. "Automatic text summarization: Past, present and future". In: *Multi-source, Multilingual Information Extraction and Summarization*. Springer, 2013, pp. 3–21.

[89] Gerard Salton et al. "Automatic text structuring and summarization". In: *Information Processing & Management* 33.2 (1997), pp. 193–207.

[90] Amani K Samha, Yuefeng Li, and Jinglan Zhang. "Aspect-based opinion extraction from customer reviews". In: *arXiv preprint arXiv:1404.1982* (2014).

[91] Jeff Sauro and James R Lewis. *Quantifying the user experience: Practical statistics for user research*. Morgan Kaufmann, 2016.

[92] Karen Sparck Jones. "A statistical interpretation of term specificity and its application in retrieval". In: *Journal of documentation* 28.1 (1972), pp. 11–21.

[93] Qi Su et al. "Hidden sentiment association in chinese web opinion mining". In: *Proceedings of the 17th international conference on World Wide Web*. ACM. 2008, pp. 959–968.

[94] Ivan Titov and Ryan McDonald. "Modeling online reviews with multi-grain topic models". In: *Proceedings of the 17th international conference on World Wide Web*. ACM. 2008, pp. 111–120.

[95] Peter D Turney. "Mining the web for synonyms: PMI-IR versus LSA on TOEFL". In: *European Conference on Machine Learning*. Springer. 2001, pp. 491–502.

[96] Peter D Turney and Patrick Pantel. "From frequency to meaning: Vector space models of semantics". In: *Journal of artificial intelligence research* 37 (2010), pp. 141–188.

[97] Bo Wang and Houfeng Wang. "Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing". In: *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*. 2008.

[98] Hongning Wang, Yue Lu, and Chengxiang Zhai. "Latent aspect rating analysis on review text data: a rating regression approach". In: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2010, pp. 783–792.

[99] Michael White et al. "Multidocument summarization via information extraction". In: *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics. 2001, pp. 1–7.

[100] Michael J Witbrock and Vibhu O Mittal. "Ultra-summarization (poster abstract): a statistical approach to generating highly condensed non-extractive summaries". In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM. 1999, pp. 315–316.

[101] Xindong Wu et al. "Personalized news filtering and summarization on the web". In: *2011 23rd IEEE International Conference on Tools with Artificial Intelligence*. IEEE. 2011, pp. 414–421.

[102] Zhijun Yan et al. "EXPRS: An extended pagerank method for product feature extraction from online consumer reviews". In: *Information & Management* 52.7 (2015), pp. 850–858.

[103] Haiqin Zhang, Zheng Chen Wei-ying Ma, and Qingsheng Cai. "A study for documents summarization based on personal annotation". In: *Proceedings of the HLT-NAACL 03 on Text summarization workshop-Volume 5*. Association for Computational Linguistics. 2003, pp. 41–48.

[104] Lei Zhang et al. "Extracting and ranking product features in opinion documents". In: *Proceedings of the 23rd international conference on computational linguistics: Posters*. Association for Computational Linguistics. 2010, pp. 1462–1470.

[105] Yu Zhang and Weixiang Zhu. "Extracting implicit features in online customer reviews for opinion mining". In: *Proceedings of the 22nd International Conference on World Wide Web*. ACM. 2013, pp. 103–104.

[106] Li Zhuang, Feng Jing, and Xiao-Yan Zhu. "Movie review mining and summarization". In: *Proceedings of the 15th ACM international conference on Information and knowledge management*. 2006.