Auto-Illustration of Short French Texts

Paula Pawlowski

UPPSALA
UNIVERSITET

Auto-Illustration of Short French Texts

Paula Pawlowski

Uppsala University
Department of Linguistics and Philology
Master Programme in Language Technology
Master's Thesis in Language Technology, 30 ECTS credits

September 21, 2020

Supervisors:
Mats Dahllöf, Uppsala University
Elena Cabrio, Inria Sophia Antipolis - Méditerranée
Pierre Kornprobst, Inria Sophia Antipolis - Méditerranée

Abstract

Text auto-illustration refers to the automatic annotation of text with images. Traditionally, research on the topic has focused on domain-specific texts in English and has depended on curated image data sets and human evaluation. This research automates text auto-illustration by uniting natural language processing (NLP) methods and resources with computer vision (CV) image classification models to create a pipeline for auto-illustration of short French texts and its automatic evaluation. We create a corpus containing documents for which we build queries. Each query is based on a named entity that is associated with a synset label and is used to retrieve images. These images are analyzed and classified with a synset label. We create an algorithm that analyzes this set of synsets to automatically evaluate whether an image correctly illustrates a text. We compare our results to human evaluation and find that our system's evaluation of illustrations is on par with human evaluation. Our system's standards for acceptance are high, which excludes many images that are potentially good candidates for illustration. However, images that our system approves for illustration are also approved by human reviewers.

Acknowledgements

Thank you to my supervisors and to Adrianna Janik.

Contents

1 Introduction 6
- 1.1 Objectives . . . 7
- 1.2 Outline . . . 7

2 Background 8
- 2.1 Image analysis . . . 8
  - 2.1.1 Image semantics . . . 8
  - 2.1.2 ImageNet . . . 10
  - 2.1.3 Image classification . . . 11
  - 2.1.4 Image retrieval . . . 12
- 2.2 Textual analysis . . . 13
  - 2.2.1 Semantic analysis . . . 13
  - 2.2.2 Named entity recognition . . . 15
  - 2.2.3 Keyword extraction . . . 16
  - 2.2.4 Corpora . . . 18
- 2.3 What to depict? . . . 18
  - 2.3.1 Data correlation . . . 18
  - 2.3.2 Event depiction . . . 20
- 2.4 How to depict it? . . . 21
  - 2.4.1 Image rendering . . . 21
  - 2.4.2 Image retrieval . . . 21

3 Methodology 23
- 3.1 Corpus building . . . 24
- 3.2 Query building . . . 25
  - 3.2.1 Keyword extraction . . . 26
  - 3.2.2 Named entity recognition . . . 28
  - 3.2.3 Dependencies . . . 29
- 3.3 Image retrieval . . . 30
- 3.4 Image classification . . . 30
  - 3.4.1 CV evaluation . . . 31
  - 3.4.2 Human evaluation . . . 31

4 Results 33
- 4.1 Corpus . . . 33
- 4.2 Development experiments . . . 33
  - 4.2.1 Query building . . . 33
  - 4.2.2 Named entity recognition . . . 35
  - 4.2.3 Dependencies . . . 35
  - 4.2.4 Image retrieval . . . 37
  - 4.2.5 Image classification . . . 37
- 4.3 Final results on the test set . . . 41
  - 4.3.1 Query building . . . 41
  - 4.3.2 Image analysis . . . 41

5 Conclusion 43
5.1 Future work 44

1 Introduction

Text auto-illustration refers to the automatic annotation of text with images. The task is usually approached with an underlying assumption or goal behind uniting text with images, which has lead the task to go by different names in the research community. Text enriching, text picturing, automatic annotation of texts, and text to scene conversion: all are applications of what this research will refer to as *text auto-illustration*. Text auto-illustration can augment understanding of text for readers with vision difficulties, low language literacy (Medhi et al., 2006), learning disorders (Alm et al., 2002), or foreign language barriers (Carney and Levin, 2002). Adding images to instructions can also help with understanding, while adding images to online texts can attract otherwise distracted internet readers.

Text auto-illustration has been a research topic in both natural language processing (NLP) and computer vision (CV). NLP is a research field focused on the interactions between computers and human languages. In part, this involves treating human language as data in order to process and analyze texts for linguistic research, with tasks focused on syntax, semantics, and discourse patterns. CV is a research field focused on computational methods that help computers understand, see, identify, and process images as humans do. In an engineering perspective, it seeks to automate tasks that the human visual system can perform.

Both fields include methods and research that are applied in text auto-illustration, but each field has approached the topic separately. NLP research has focused on identifying which parts of a text are most relevant to text meaning and pictoral representation. This is usually addressed with labeled data sets specially curated for the task. CV research on the topic has often dealt with image annotation, or mapping words to images, in order to label image data for future research use or for image captioning. When NLP and CV research has been combined, the point of departure has usually been from images to natural language. The classic and most common task is image captioning, or generating textual descriptions of images, where words are matched to images.

This research will attempt to unite CV data and tools with NLP research in the less common task: matching images to words. Our observations about text auto-illustration motivate the three research questions underlying our research objectives.

First, we observe that research on text auto-illustration is dominated by publications based on English texts. We want to investigate if we can use existing multilingual NLP tools to develop a French text auto-illustration system.

Secondly, text auto-illustration implicates research methods in both CV and NLP, but the topic has been stuck in "research silos" in both fields. We want to frame text auto-illustration as a task that unites NLP and CV research to augment text with images. Practically, can we use NLP text analysis to verify the results of CV image labeling and classification results?

Finally, we see that text auto-illustration research uses curated data and human evaluation, which are both expensive and belabor research. Can our pipeline's results be automatically evaluated without sacrificing quality?

1.1 Objectives

This master thesis project has two main objectives:

- Build an unsupervised system for text-auto illustration of short (200-500 words) French texts. This is novel as most research is applied to English texts and there is currently no publication focused on auto-illustration of French texts.
- Compare human evaluation of texts' illustrations with CV evaluation. We will analyze retrieved images with models trained on ImageNet data (Deng et al., 2009). We compare how the image evaluation using CV algorithms compares to human evaluation. As most text auto-illustration research is evaluated by humans, developing a reliable and automated evaluation system of retrieved images would facilitate further research in text auto-illustration.

We carry out our research as part of an internship in INRIA (INRIA, 2019), where there was interest in illustrating texts for elderly readers with vision impairments. Patients requested the illustration of short and descriptive texts, namely those related to travel. The addition of images was intended to aid word association for readers who were native and fluent French speakers and readers.

1.2 Outline

The research will be presented as follows. In Chapter 2, we introduce the background of auto-illustration. We begin with image analysis and applications of NLP research, followed by textual analysis. The latter part of the chapter presents the different ways of approaching text auto-illustration. In Chapter 3, we present our methodology. We present the three main parts of our approach: corpus creation, query creation, and image retrieval and analysis, along with our standards for the human review. In Chapter 4, we present the results of our development set and test set. Our presentation of the development set results also include an explanation of our final choices for the the test set. Finally, we conclude with Chapter 5 with a discussion of our results and their implications.

2 Background

Text auto-illustration is one concrete task that unites NLP and CV research, but the two fields are fundamentally linked: both are pillars of artificial intelligence (AI). There are many ways to branch and categorize the AI field, especially as technologies wearing its label have weaved into contemporary life. Norvig and Intelligence, 2002 use the Turing Test as a reference for what is core to the field and what a computer must possess to demonstrate "a satisfactory operational definition of intelligence." A computer needs to possess capabilities in natural language processing, automated reasoning, and machine learning. To possess these capabilities, a computer will also need computer vision and robotics. As such, text auto-illustration brings together core components of AI.

The growth of CV and NLP within AI has been paired with the booms in first machine learning, and now in deep learning. More intuitively, the connection between NLP and CV is as practical as the relationship between what humans see and how we use language to describe what we see.

In this chapter, we introduce the principles of CV and NLP research. We assume the reader has a NLP background, so we reference basic NLP concepts in Section 2.1, even though we dedicate Section 2.2 to a presentation of NLP methods. In these first two sections, we present each field's basic methods and tools that we integrate in the methodology we develop in Chapter 3. In the final two sections, Section 2.3 and Section 2.4, we present the main theoretical questions we ask before designing our system: What to depict? How to depict it?

2.1 Image analysis

Computer vision tasks can be summarized by the concept of the 3 Rs: *recognition* of objects in an image, *reorganization* or segmentation of raw pixels into groupings that represent an image, and *reconstruction* or creation of images. With the recognition field, core CV tasks are *image classification* and *object identification.* The results of these tasks, the labeling of images and identification of objects, are expressed in terms of natural languages. That is, researchers interpret results with labels that are in turn interpretable by NLP methods.

Technically, we can also frame NLP and CV tasks in a similar way. Both NLP and CV represent the research object, text and images respectively, as data. This data can be represented in vector form to work with machine learning and deep learning algorithms. The vectors are numerical representations of *features* that are identified in the text or image. In the following subsections, we describe with greater detail image analysis and the basic CV tasks that have potential with NLP. We present examples outside of text auto-illustration to contextualize our first objective of uniting CV and NLP methods in our research.

2.1.1 Image semantics

In the context of images, a semantic feature describes the visual content of an image, from low-level features such as color, gradient orientation, or the spatial layout of a scene. Just as POS tagging is often a sub-task of semantic text analysis, identifying the "parts" of an image helps build an understanding of the image semantics: "Semantic

segmentation is a natural step in the progression from coarse to fine inference" (Learning, 2019). A word may have more than one sense which can be inferred by noting its co-occuring words, while a pixel in an image may appear differently according to scale and image quality. The process leading up to semantic image segmentation can be divided into the following steps:

- **Classification**: making a prediction for the whole input (image)
- **Localization/detection**: identifying classes within the input, including their location
- **"Dense" predictions**: fine-grained inference by labelling every pixel, so that each pixel is labeled within a regional class (Learning, 2019)

The image classification models that we will implement will follow these steps, "out of the box." Assigning the labels to image regions in object classifiers is standard practice in CV (Del Pero et al., 2011). When faced with bad or incorrect classifications, we can assume that the cause is noisy data, such as images with low quality, or training gaps.

Another source of bad labels is the fact that traditional classifiers focus on features of smaller groupings of pixels, which do not always correlate to the semantic features as identified by humans. This gap between automated results and human evaluation is referred to as the *semantic gap* (Ma, J. Zhu, M. R. Lyu, et al., 2010). One way to deal with the semantic gap is to incorporate NLP methods between the automated results and human evaluation. For example, an image is usually associated with some sort of text. When visual features are compared to one another, the visual similarity of the images can be combined with the similarity of the texts associated with them (Santini, 2001). Human evaluation can then be based on the semantic similarity of the image and its text context. Likewise, in cases where classifiers identify multiple objects with semantically similar text labels, word properties such as synonymy can used to find a hypernym, or more general term, that best describes an image without oversegmenting (Cao and Fei-Fei, 2007). This precedent of NLP and CV methods being used to improve image classification and semantic pertinence is relevant to our second research objective of automatic evaluation, and to the spirit of our research goals.

Image feature detection

Just as we parse a text to extract text features, different algorithms exist to parse an image for feature detection. The scale-invariant feature transform (SIFT) algorithm is used to detect and describe local features in images (Lowe, 2004). SIFT is relevant to distinguish because it is fundamental to image classification. The algorithm disambiguates image pixels by a series of operations that manipulate an image to identify consistent keypoints. The algorithm is focused on detecting *keypoints* of *interest points*. Interest points are locations in an image that indicate a spatial location, or an area where an object to be identified is located in a photo. The keypoints are indicators that an area of interest is beginning in an image (Kelman et al., 2007). For text auto-illustration, understanding what a keypoint is and how it is identified can later inform our methodology choices of which image classifier to use.

To identify they keypoints, major tasks include:

- **Scale-space selection**: blurring and resampling an image to create the best scale space, to ensure scale invariance
- **Blobs/interest point localization**: using a series of Gaussian methods to detect points of interest

- **Orientation assignment**: find the keypoints to ensure rotation invariance, so that an object's position does not distort it from proper classification

These steps are classic parts of a CV keypoint detection pipeline. The research of Kesorn et al., 2011 has used the concept of a *visual word representation* to include NLP feature vectors in an image classification pipeline. Feature vectors from training data are clustered and associated with a token called a "visual word." The visual words in an image region are compared from semantic similarity to detect where a keypoint may be located. Other approaches use predicted visual words to find, rather than define, keypoints (Sudderth et al., 2005).

Keypoint detection, and the feature detection that follows, are carried out with a vector representation of the features that are extracted from the images. After the algorithm is performed, the output is the features that are relevant for the image. In both NLP and CV, we leverage *low-level* and *high-level* features. In CV, low-level features include identifying "edges" and "blobs," or the base groupings of pixels that compose images, while in NLP low-level features refer to part-of-speech (POS) identification and tagging. High-level features in CV include identifying objects and events, while in NLP higher-level tasks similarly focus on identifying larger units of text, such as named entity recognition (NER). Even though these fields treat different mediums, we can talk discuss tasks using similar vocabulary (Barnard, 2016).

To this extent, NLP can also help orient CV tasks. Scene type identification is one type of image classification done on the basis of feature detection (Serrano et al., 2004). A scene within an image can be labeled as "outdoor" as opposed to "indoor." We may want a more fine-grained scene label, such as "mountain range" or "forest." This can be considered a supervised learning problem, where a better image data set with more labelled scenes can help CV models produce a better label. However, incorporating NLP methods, a CV object classification task can reject object labels that are not semantically related to "outdoor." A fine-grained scene label of "kitchen" would be rejected because it is not semantically similar to "outdoor."

This methodology can be carried over to other CV tasks. For example, we may also want to identify an object within the image. If one CV model labels a scene as "outdoor" then an NLP model can help reject object labels in the scene that are not semantically similar to "outdoor." For example, an object label of "kitchen table" can be reject because the word is more semantically similar to "indoor" than "outdoor." This example incorporates NLP methods to improve CV model classification results, which our research does not aim to do. Our research will incorporate NLP methods in analyzing the results of CV models, but using textual semantic analysis with image feature detection informs our methodology presented in Chapter 3.

2.1.2 ImageNet

Research in image semantics depends on large, labelled data sets. Today, the most significant data set and resource is ImageNet (Russakovsky et al., 2015), a collaborative project that aims to create a standardized data set for image research. Due to the acceptance and popularity of the project, ImageNet has also become an important data set for not just CV research, but deep learning and AI research in general (Deng et al., 2009). ImageNet is associated with the annual Large Scale Visual Recognition Challenge, or ILSVRC, which is an annual competition that uses subsets from the ImageNet database to develop techniques and present benchmark results in AI research and state-of-the-art algorithms.

The project's contribution is a large-scale database aimed at labeling and categorizing images based on a defined set of words and phrases organized according to the WordNet

hierarchy (Miller et al., 1990), which we describe in Section 2.2.1. The project consists of over 22,000 categories and over 14 million images. Each node of the WordNet hierarchy is depicted by at least a thousand human-annotated, quality-controlled images per synset, a set of cognitive synonyms that represent one meaning (Miller et al., 1990).

These images are annotated with the image contents, expressed as a WordNet synset, with some images annotated with fine-grained details. Table 2.1, outlines the most common categories in ImageNet. The first five places represent the most well-represented categories in ImageNet, while the remaining categories are those that are assumed to be semantically linked to texts that will used in our implementation.

Since ImageNet is designed to work for high-level deep learning development, the data set is curated for fine-grained identification tasks. Some WordNet categories are seemingly over represented, but this is presumably intentional. For example, the largest category is plants, which is a category that lends itself to many fine-grained tasks as plant varieties are part of taxonomies that are well-documented.

rank	High level cat.	Synsets	Avg. images per synset	Total no images
1	plant	1666	600	999K
2	person	2035	468	952K
3	structure	1239	763	946K
4	mammal	1138	821	934K
5	bird	856	949	812K
10	tree	993	568	564K
8	flower	462	735	339K
10	animal	3822	732	2799K
26	geological formation	151	838	127K
27	food	1495	670	1001K

Table 2.1: Examples of categories in ImageNet

ImageNet is significant in the CV research community for creating a benchmark and gold standard for research and communication. Each year, the conference ILSVRC results in publications and discourse in CV, but also allows the CV community to integrate fundamental NLP resources, namely WordNet and semantics, into their research. From building image classifiers based on WordNet classifications used in ImageNet (Yu et al., 2015) to evaluating unsupervised cross-modal (image, text) disambiguation (May et al., 2012), ImageNet is integral to CV research, as it is to our research.

2.1.3 Image classification

In the previous sections of this chapter, we have presented CV tasks, how NLP can be applied to them, and how ImageNet is significant for research. Image classification is central to the text auto-illustration pipeline that we create. The previous section described its technical details. In this section, we will describe the image classification architectures we will use to implement image classification in our pipeline.

MobileNet

MobileNet (Howard et al., 2017) is an architecture designed by Google and intended for mobile devices. It uses a convolutional network, which is computationally expensive and demands resources. Usually CV models become more accurate with adding more layers and becoming a more complicated network, but this has costs for size and speed.

MobileNet cuts down the size and speed of classification by using depth-wise separable convolutions to build lighter neural networks. This is done by "filtering" channels and filters and splitting separate layer for filtering and a separate layer for combining. For the purposes of our research, a model that is accurate but not computationally expensive would be beneficial. Since a given text has many entities, we prefer a model that is fast but accurate.

ResNet

ResNet stands for deep Residual Network (He et al., 2016). Its core innovation is using the idea of a "skip connection" to avoid the "vanishing gradient problem" in neural networks. Deep learning is based on the idea that layers encode information that can be feed-forwarded. To avoid over-fitting data, more data, and more layers, are added to a model, which can be referred to as a "deeper" network. However, as layers are added the gradient is back-propagated to earlier layers and the deeper layers may start degrading (Dwivedi, 2019). By using what is called a "skip connection" the ResNet model stacks some layers but not all: low performing layers are bypassed in mapping. The goal is to make sure that a higher layer will perform better, and not worse, than its previous layer. ResNet won the ImageNet 2015 competition, which makes it interesting for our purposes of leveraging the ImageNet labeled image sets and WordNet labels with our own documents.

2.1.4 Image retrieval

The images used in our pipeline will be accessed from external resources. Retrieving, or searching for, images can be divided into two different approaches: content-based and context-based image retrieval.

Content-based image search

A content-based image search searches the contents of an image, rather than metadata, such as keywords, tags, or descriptions associated with the image, to make a match. "Content" refers to the image's contents that can be analyzed using CV techniques, such as colors, shapes, and spatial orientation.

Query-adaptive image retrieval (QAIR), or query by example (QBE), is closely connected to content-based searches (Qin et al., 2013). With QAIR, features are extracted and stored from an example image to which retrieved images are then compared. The querying therefore depends on what features are used. Some image features can be used to match those features that humans are most perceptive to, while other features can be chosen based on the computational performance of algorithms.

The semantic gap, or the inconsistency between visual and semantic similarity, is a problem of content-based searches (Ma, J. Zhu, M. R.-T. Lyu, et al., 2010). Even if a retrieved image matches the features of the example image, the retrieved image is not guaranteed to be a semantic match of the example image or query. This motivates our reasoning that text auto-illustration cannot end at image retrieval.

Context-based image retrieval

Context-based image retrieval (CBIR) is also referred to as "concept-based" image retrieval (Westerveld, 2000). Information about an image's contents can be deduced from the image's context and associated metadata, such as where the image appears, how it is displayed, and who took the picture.

When it comes to the quality and accuracy of retrieved images, this approach is sensitive to the *paraphrase problem* (Westerveld, 2000). Concepts can be discussed using different words, or the same words can describe different concepts. This potential for textual ambiguity also coexists with ambiguity in image choice. Images that are chosen to accompany text do not necessarily correlate with the text's contents, but may be chosen to highlight an element of the text or are chosen because they are visually pleasing. For tagging or keyword annotation, images may be tagged with terms that are popular, trending, or have high search engine ranking. CBIR is less expensive than QAIR, and more common, but requires some sort of filtering and evaluation, especially if retrieved images will be used to illustrate a text.

2.2 Textual analysis

We explain our methodology in Chapter 3, but our background discussion of textual analysis is motivated by the *query* that we will build and its components. In this section, we will present NLP tasks and methods that we will implement in our text auto-illustration pipeline. In doing so, we will refer to "basic" NLP concepts. "Basic" features, or linguistic features that are fundamental for NLP tasks, are not to be confused with "simple" features. Accurate extraction of low-level linguistic features remains an active research field.

"Tokens"refer to textual units that are commonly referred to as "words." For the text analysis and linguistic features that will be described, part-of-speech (POS) labelling and dependency parsing are pertinent building blocks of high-level linguistic identification (Jurafsky and Martin, 2000). POS labelling refers to assigning a grammatical POS to a token. Text is entered into pre-trained part-of-speech classification models and tags are assigned to tokens. Dependency parsing describes the relationship between words in a sentence. The goal is to build a tree that assigns a single parent word to each word in a sentence. The root is usually the main verb in the sentence. Parsers using dependency relations have performed well for analytic languages, such as English and French, which are not heavily inflected (McDonald et al., 2013). In presenting semantic analysis (Section 2.2.1), named entity recognition (NER) (Section 3.2.2), keyword extraction (Section 3.2.1), and corpora (Section 2.2.4), we will often reference these basic NLP concepts.

2.2.1 Semantic analysis

Semantic analysis assigns language-independent meaning to words. This analysis can be done on a word-by-word, sentence, or document level. A word-by-word analysis refers to the meaning of a word whereas a semantic analysis on a sentence or document refers to the topic of a text. Semantic analysis is complicated by *polysemy*, meaning that one token may have multiple meanings. These meanings are commonly referred to as *senses*. The classic example is *bank*, which has at least two common senses, referring to a *river bank* and a *financial institution*. For text auto-illustration, it is necessary to disambiguate polysemy and identify a word's intended meaning in order to correctly illustrate a concept. WordNet and topic modelling can help us achieve a correct semantic analysis and illustration.

WordNet

WordNet (Miller et al., 1990) is the dominant resource in computational research on semantics. It is a lexical knowledge resource in which entries are organized according to a concept. For example, the meaning, or concept, of *the feeling that comes when*

something burdensome is removed or reduced would be assigned as the first sense of "relief," $relief^1$. Lexical entries referring to the same concept are paired in a *synset*. The synset of the "relief" concept would be $relief^1$, $alleviation^1$, $assuagement^1$. However, the synset of the concept *sculpture consisting of shapes carved on a surface so as to stand out from the surrounding background* would include $relief^9$, $relievo^1$, $rilievo^1$, $embossment^2$, $sculpturalrelief^1$. WordNet notes that "relief" is a token that is attached to more than one synset, or meaning.

Synonymy, as expressed through these synsets, is the core of WordNet's structure, but there are also other relevant relationships. Multiple senses of a word can be differentiated by their taxonomic relations. These relations outline words' relationships with each other and allow us to interpret word senses. They apply to nouns, but the analysis of verbs is similarly structured. Adjectives' relations are structured in terms of antonymy.

Synsets are related to other synsets through super-subordinate relations, but there are a total of 12 semantic relations between synsets, such as *is-a* and *part-of*. Through the hypernomy relation, each noun's hierarchy ultimately goes up to the root node of *entity*. Relations relevant to our research include:

- **Superordinate relation**: A hypernym is a word in a larger class than another word. *Dogs* is a hyponym class of *all walking dogs*.
- **Subordinate relation**: A hyponym is a word that is subordinate to another. *All walking dogs* is a hyponym class of *dogs*.
- **Part-to-whole relation**: A meronym is a component of a larger whole, that can represent the whole semantically. *Paw* is a meronym of *hind leg*.

Computational and linguistic resources for semantic analysis use these relations to define senses and to compare research. For our research, these relations are important as they help us semantically analyze texts and to use other NLP and CV resources that depend on semantic analysis and use WordNet's labels.

Topic modelling

Taxonomic relations focus on word relations, but semantic analysis is also applied to entire texts or documents. Since our text auto-illustration pipeline will illustrate short texts, and not just individual tokens, identifying a text's topic may be useful to ensure that illustrations are relevant.

Taxonomic relations between words end up being foundations for tasks within information extraction, which is the process of extracting information from *unstructured* data. Text which is unstructured has not undergone processing or analysis in the way of syntactic or semantic analysis. Topic modelling approaches generally rely on probability distributions, data clustering, and machine learning algorithms. The common output of topic modelling algorithms is a distribution of topics that describe a text, but related tasks also include text segmentation and keyword extraction (Blei et al., 2003).

Text segmentation refers to topic modelling on a sentence-by-sentence level to segment text by topic and similarity. Although written texts may seem "structured" by way of their organization into paragraphs and sentences, such formatting may be superficial from a computational perspective. Common approaches to segmenting text include comparing sentences based on similarity measures to segment a text into segments divided by a common topic. Text segmentation, through topic modelling, may be relevant for the coherent presentation of an auto-illustration.

2.2.2 Named entity recognition

The previous section introduced semantic analysis and a task related to taxonomic relations. Word sense disambiguation (WSD) is a subtask within semantic analysis. The task is to identify which sense a word has in a given text (Jurafsky and Martin, 2000). Algorithms identifying senses often rely on the collocations, or immediate neighbors, of words, with the assumption that the senses of a word's neighbors help identify the word's sense. This is usually determined through pattern-based approaches, which depend on POS or syntactic analysis, or supervised machine learning methods, which depend on labeled training data.

A subtask of WSD is *entity recognition* and *named entity recognition* (NER). Entities are words or phrases that refer to defined categories, such as people, organizations, places, or concepts. Named entities usually exclude concepts, and refer to predefined entities that are consistently associated with a name. For example, *bank* is an entity or concept, but *World Bank* is a named entity.

In our research, we start with the assumption that identified entities are good candidates for auto-illustration. Key semantic components of our text auto-illustration approach will focus on WSD and NER. We use Babelfy (Moro et al., 2014), a graph-based approach to entity linking and WSD, for our knowledge extraction. The semantic analysis that leads to the WSD is piped in through BabelNet (Navigli and Ponzetto, 2012), a multilingual semantic network. BabelNet links the semantic lexical resource of WordNet (Miller et al., 1990) to Wikipedia (Wikipedia, 2018b), which is used as a multilingual encyclopedic resource. Babelfy's approach allows us to tackle WSD and NER in one concurrent process, while also allowing us to pursue a multilingual approach for our French texts.

In the next subsection, we will describe the tasks of WSD and entity linking in BabelNet's multilingual semantic network and how it is utilized in Babelfy.

BabelNet

WordNet's lexical information and structure is part of BabelNet's resources. In this subsection, we describe BabelNet's structure to justify why it is important part of our non-English objective.

Along with WordNet, Wikipedia is core to BabelNet. Wikipedia's information is loosely structured and is not always consistently labeled, as it is a collaborative online project, whereas WordNet is a result of formal research within academic institutions. Wikipedia's structured information is found in info boxes, which summarize attributes of an entity that is the subject of a page. WordNet's structured information is organized through explicit relations. Likewise, Wikipedia's redirect pages serve as parallels to the synonymy relations in WordNet: WordNet numbers word senses, while Wikipedia provides a disambiguation page. For example, the *relief (disambiguation)* page states that the default *relief* refers to a sculptural technique, while *relief (emotion)* is presented as "[another] common meaning" (Wikipedia, 2018a). Both WordNet and Wikipedia have a relevant data and structure for WSD, and both have multilingual support. WordNet has explicit relations based on lexical properties, while Wikipedia contains related entities. Both can be considered graphs and are language-independent. For WordNet, a word sense is a node with semantic relations to other senses as edges. For Wikipedia, a Wikipage is a node connected to other pages with URLs as edges.

BabelNet combines WordNet and Wikipedia, taking advantage of the explicit and structured relations of WordNet with the supplementary information to Wikipedia. A Babel synset is the union of a WordNet synset to which a term belongs to and the inter-language links. For terms that are missing inter-language links in Wikipedia, a machine

translation is made, which is then verified through different steps. This approach has proven successful in addressing multilingual needs that were not addressed by earlier tools and BabelNet has gained widespread acceptance and use in the semantic community– a motivation to its integration in our pipeline.

2.2.3 Keyword extraction

Keyword extraction aims to concisely describe contents of a text with one word or phrase (Firoozeh et al., 2020). Our goal for keyword extraction is to find the best keyword that will serve as a thematic keyword for each query. In the context of our research objectives, a good illustration of a text should not contain random images related to a text, but images that are coherent. Determining a text's keyword, or a *thematic keyword*, gives us a context for a document and links all images used for a text's auto-illustration. In this section, we will present some implementations of keyword extraction that we will experiment with to find a document's keyword.

Statistical methods

Statistical methods represent a group of methods that use term frequency counts and document statistics to find keywords.

TF-IDF A basic approach for keyword extraction is using a frequency criterion to select the important keywords in a document. *Term frequency and inverse-document frequency* (TF-IDF) is a fundamental method in text mining to measure how important and relevant a word is in a document (Jones, 1972). It is a frequency based method, which generally yields poor results (Mihalcea and Tarau, 2004), but is useful as a base indicator. IDF accounts for common, or frequent, words. If a word occurs in each document, it likely does not bear special meaning to each document. How to calulate TF-IDF is described in Figure 2.1.

TF-IDF = TF(term in a document) * IDF(term)
TF(term) = term frequency in a document/total number of terms in a document
IDF(term) = log(total number of documents/number of documents containing the term)

Figure 2.1: TF-IDF calculation

TF-IDF's simplicity and intuitiveness is an advantage. Its results returns a vector per word per document based on frequency, which can be interpreted by humans. For texts which explicitly note the subject, the TF-IDF may perform as well as more complicated methods. However, for texts which describe a topic, without naming it explicitly but simply describing it, TF-IDF may be ineffective in identifying the true subject.

LDA The LDA (latent Dirichlet allocation) algorithm is generally used for topic modelling (Blei et al., 2003). While keyword extraction aims to concisely describe contents of a text, topic modelling aims to find the topic of the text. There can be overlap because a keyword can be the same as a text's topic; a topic can be considered a collection of dominant keywords.

LDA is a generative unsupervised probabilistic algorithm which identifies the top K topics in a data set as described by the most relevant N keywords. Word order and sentence order in the documents are not important. The algorithm is a clustering algorithm where the number of possible K topics, or clusters, is predetermined. The

algorithm initially randomly assigns a word to a cluster. For each word in a document, the algorithm computes two things: 1) the proportion of words in document *d* that are shared with the other documents in the topic cluster and 2) how many documents containing the word have been assigned to the topic. The word is then assigned to a new topic based on the product of these two probabilities. The process is iterated until convergence.

Graph-based methods

Graph-based methods represent a group of methods that use graph based ranking algorithms to find keywords. Mihalcea (Mihalcea and Tarau, 2004) explains that applying graph based ranking algorithms to natural language texts consists of:

- Identifying the units at hand.
- Identifying the relations that connect the chosen units.
- Iterating the graph-based ranking algorithm until convergence.
- Sorting vertices based on their final score.

TextRank TextRank (Mihalcea and Tarau, 2004) is inspired by PageRank, the algorithm used by Google search engine. The basic idea of the algorithm is that some sort of lexical relation between two words is used to connect them. In the original implementation, the co-occurrence relation is used (whether tokens are contiguous): two words are connected if they appear within a window of maximum N words (N being 2 to 10 words). Each word within the document is a node. The nodes, or vertices themselves, can also be filtered with stop words or with POS filters. The algorithm is also versatile in that in can be used for words, collocations, or sentences.

Each node is scored based on the number of times a word co-occurs with words. The weight is based on the product of the sum of words the candidate follows (its predecessors) and the inverse ratio of words it precedes (its successors). This product is then smoothed by the *damping factor*, or the probability of going from one vertex to another in the graph. The weights of the nodes are propagated until the scores converge. Once the algorithm has converged and the scores have stabilized, the words are sorted and a fraction of the top words retained for post-processing. If any of the keyword candidates are contiguous in the original document, they are merged and the multi-word keyword is returned.

RAKE The RAKE algorithm (Rapid Automatic Keyword Extraction) (Rose et al., 2010) is domain independent: it does not depend on a labeled corpus, which is an interesting detail for our goal to reduce expenses. A document is split in to an array of words that are broken at word delimiters. The words are then split into sequences of contiguous words where each sequence is further split at stopwords.

A co-occurrence graph is built to identify the frequency that sequences are associated with each other. These final substrings are the keyword candidates. The score of each candidate is *degree(word)*/ *frequency(word)*. The degree is based on how frequently a content word co-occurs with other candidates. The more times it appears in other candidates, the higher the degree is. Intuitively, this means that the more times the word appears within candidates, the more likely it itself is a potential keyword candidate. Dividing by the total frequency smooths for words that are simply common words, but do not appear in longer substrings. To determine which substring is the keyword, the score of each word in a substring is added.

2.2.4 Corpora

In the previous subsections, we have described some NLP tasks and tools that are relevant to our textual analysis: WordNet, NER, and keyword extraction. In this subsection, we will describe the texts themselves. Text auto-illustration research largely relies on task-specific corpora. The corpora for text auto-illustration as a learning and work aid in Agrawal et al., 2011, Mihalcea and Leong, 2008, and Coyne and Sproat, 2001 are likewise based on the texts that the intended audience would use. VizStory (Huang et al., 2013) uses a corpus of fairy tales because of the known structure of short fairy tales: a sequential unfolding of events with consistent characters.

There is no example of similar, task-specific corpora for French research as there is no published research for French-language based text auto-illustration. English-language resources dominate NLP research and development and are the reference point for data sets and methods, but French NLP research is active (Hernandez et al., 2010). Accordingly, there are major corpora frequently used in French-language based research: frWaC, French Web Corpus (frTenTen), and the FR-Wikipedia corpus (Baroni et al., 2009). *frWaC* corpus is a project of the Web-As-Corpus Kool Yinitiative and is based on websites with the .fr domain and contains 1.3 billion medium-frequency words from the Le Monde Diplomatique corpus (Ferraresi et al., 2010). French Web Corpus (frTenTen) is another corpus based on a crawl of Internet texts and contains over 10 billion words from different French varieties. Adding to Wikipedia-based resources, the FR-Wikipedia corpus is based on a 2008 archive of Wikipedia articles.

None of these major French corpora respond to our needs, so we create a data set for our research purposes. Creating a custom corpus allows us to choose a topic domain that will contain text that can be auto-illustrated. It also allows us to control for text length and resources. Finally, as this research was a part of an INRIA research project, feedback from research participants also motivated this decision.

2.3 What to depict?

Having presented the basics of NLP and CV concepts, we can discuss the first question of text auto-illustration: What to depict?

What is depicted in a text affects its interpretation. As a marketing tool to attract readers, text auto-illustration should highlight a text's trendy or intriguing elements. As a learning aid, text auto-illustration should highlight objects that will aid in text understanding. A text's audience can instruct what is portrayed. How a text is analyzed can also instruct what is portrayed. Different approaches to analyzing text data imply different methods. The NLP community research can be generally divided into two approaches towards analyzing text for data for auto-illustration: *data correlation* and *event depiction.*

2.3.1 Data correlation

From the perspective of data correlation, text auto-illustration is an information retrieval problem. In this approach, texts contain data points that need to be matched with images referring to the same data. Text auto-illustration is a question of matching a set of data within one media, text, to its matching counterpart in another media, images. This approach assumes that the most relevant or important data, based on some statistical measure of relevance, will also be most visually potent. In data correlation, these presumably relevant terms are *picturable objects* in a text (Krawczak et al., 2016).

Within *picturability* data correlation approaches, one method is to focus on a part of speech to portray. Often, this assumes that nouns or named entities have a direct

correlation with an object in the real world. Mapping a concept such as "tree" to an image is not without debate as "tree" is an approximation of a concept. However, a conventional agreement of the visual form of a "tree" can be agreed upon. Picturability methods are often applied to textbook illustration (Agrawal et al., 2011) or simple sentence representation geared towards young learners (Mihalcea and Leong, 2008). An approach in the WordsEye project (Coyne and Sproat, 2001) illustrates a text scene by adding objects identified in a text, but it is targeted towards graphic designers, functioning as a work-aid more than a learning-aid.

For methods that do not want to assume picturability based on a part of speech, some sort of picturability measure can be established. Many texts contain abstract concepts (politics, friendship, or happiness) that cannot be readily mapped to a tangible object in the real world. Depicting these concepts requires debating how to correlate these abstract notions to a single image. To avoid dealing with these concepts, X. Zhu et al., 2007 use an approach of scoring the picturability of words with a picturability logistic regression model was trained on a manually-labeled set of 500 words, randomly selected from a large vocabulary.

More comprehensive methods combine linguistic analysis and some sort of picturability measure. A possible approach is using image annotations, rather than image features, to compare similarity of texts and an image. The work of (Itabashi and Masunaga, 2005) approaches text auto-illustration as a matter of information correlation. In this approach, sentences are expressed as vectors whose elements represent the index-term weights. A web image retrieval is based on the similarity of image annotations and text. This approach relies on lexical features of input text and image annotations, assuming that shared word tokens between the two medias correlate to shared meaning. This approach necessitates certainty about image contents and annotation consistency.

Notable work

Most research combines this knowledge of easily identifiable objects, abstract concepts, and information retrieval. Semantic understanding of text, or the meaning of text, can also be used as a text *feature*. An example is VizStory (Huang et al., 2013) which combines the approaches to picturability to develop a text auto-illustration algorithm.

In the VizStory system, fairy tales are illustrated with retrieved images based on the idea of keywords and semantic understanding. In the approach, a text is segmented and three sets of keywords are used to form an image query.

- A *theme keyword* is extracted from a segment and leading keywords from the text. The theme and leading keywords are used to retrieve web pages, which are then clustered according to similarity.
- A set of *context keywords*. The most frequent words in each cluster of web pages are compared to the text's lexicon. The union makes up the context keywords.
- An *event keyword*. An event query captures the semantic relationships in the sentences in relation to a verb, namely the subjects and objects. If the subject and object are also captured as keywords in the theme keyword, the verb is included in the final query.
- Images are retrieved through a Google image search, with filtering done based on the lexical content of the web pages from which the image is retrieved.
- Results are evaluated by humans.

VizStory, along with the other data correlation approaches, uses text and the data it contains, such as the linguistic features and semantic details extracted from the text, to correlate them to images that share data. VizStory's approach can be adapted to different languages and domains and serves as an inspiration for this research.

2.3.2 Event depiction

Text auto-illustration can also be considered a way to depict and recreate scenes expressed in text. Overall, the event depiction approach is often used when text auto-illustration is researched within an AI context. The term "text-to-scene" conversion is popular within the AI research community (Coyne et al., 2010). Semantic frames can be modeled as logical propositions which can be fed into a system. This can also be implemented within question-answering tasks. The generated image can be used within a question-answering task to verify an AI system's performance in reasoning about a scene that is created based on initial text (Kunda, 2018).

Unfiltering

One approach is to incorporate words that are filtered out in noun-based approaches. Research such as (Zitnick et al., 2013) focus on literal depictions of scenes by using prepositions, verbs, and adjectives to determine picturability. These words are not necessarily *content-bearing*, but are *content-orienting*. They do not contain content that can be portrayed in images, but they help understand a text in terms of its semantic components. Such text does not describe an "event," with actors and actions, but is declarative. In this case, an event depiction approach would include the prepositional phrases that describe the relationship of the words described.

Semantic frames

Other event depiction approaches include semantic data and *semantic frames*. Semantic frames refer to the perspective or participants of an event. A sentence can be interpreted as depicting an event that can be expressed in terms of its participants and their actions. Understanding a sentence in the context of semantic frames requires that the meaning of a single word cannot be understood without also considering surrounding words (Fillmore and Baker, 2010). An event like a dinner can be explained in terms of verbs like "eat" from the perspective of a diner, or "make" from the perspective of the chef. For sentences which do not have clearly defined actors and objects, the notion of semantic frames can help disambiguate words. Disambiguation refers to identifying the correct meaning to words. Contrary to the data correlation approach, this event depiction approach does not assume that content-bearing tokens will correlate to powerful visual aids. The event depiction approach depends on tokens which are not content-bearing but are content-orienting.

This approach is especially useful for text auto-illustration which aims to depict instructions or a process. (Johansson et al., 2005) research involves illustration of traffic reports. In order to visualize and recreate traffic incidents, reports are interpreted in terms of semantic frames and the relation of objects: a clear example of event depiction. Sarma et al., 2018 research illustrates exercise texts to accompany instructions with animated images. The research set-up employed a semantic parser to label actions, body parts, and location.

2.4 How to depict it?

After, "What to depict?", we ask "How to depict it?" Be it with photographs, illustrations, or graphics, there are two ways to procure images for depiction: *image rendering* and *image retrieval*.

2.4.1 Image rendering

Image rendering refers to creating custom images to depict a text. Image rendering is most closely associated with auto-illustration techniques that focus on event depiction (Barnard, 2016). Because semantic frames and semantic role labeling identify actors, subject, and objects, an approach that uses them has to ensure a depiction that is loyal to the interplay of the actors. Since this can mean capturing fine-grained details, approaches that focus on semantic frames use image rendering. The WordsEye project was a sort of rare hybrid: 3D scenes were created by putting together existing and labeled 3D object models and positioning them according to semantic representations in the text.

A common use of image rendering is for auto-illustration of instructional or informative text. Rendering images ensures that instructions are accurately represented. The research cited in Section 2.3.1 all illustrated text with custom rendered graphics or illustrations, such as in Johansson et al., 2005. In general, image rendering is an expensive and work-intensive approach. Graphic illustrators can automate the process, but the approach requires human supervision and intervention, especially for use cases which need to have verified image contents.

2.4.2 Image retrieval

For use cases where image rendering is impractical or expensive, text can be illustrated using retrieved images. Image retrieval entails finding images from existing resources, be it databases, corpora, or the internet. This approach usually assumes that images are annotated with tags, captions, or some sort of textual information that describes image contents. Making *queries* to find these annotated images is central to image retrieval. Queries based on the text are created to retrieve images. How to create the query depends on what should be depicted and the resources that are queried.

Labelled databases have the benefit of being reviewed and tagged for use. Their main disadvantage is that they are labor-intensive as new images and tags are added by humans. Although maintained databases may contain less noisy data than an internet search, they are also often limited in terms of domain and tags, which risks a retrieval selection bias.

Little research uses an open internet search, with most query-based approaches favoring some sort of curated environment. In the works on data correlation cited earlier, Agrawal et al., 2011 presents a compromise between the database and open-internet approaches: images are retrieved from Wikipedia articles. In this way, the image search had access to Wikipedia's large corpus of data, with the added benefit of Wikipedia's metadata. UzZaman et al., 2011 and Lau et al., 2006 have also used Wikipedia as an image source, and the Wikipedia Image Retrieval Task has also been a reoccurring task at the ImageCLEF conferences on multimedia information retrieval (ImageCLEF, 2019).

Photo hosting internet sources, such as Flickr (Flickr, 2019) and Unsplash (Unsplash, 2019), are frequently used as they do include user tags and metadata that can be used to refine searches (Li et al., 2008), (Delgado et al., 2010). It is common practice to use Flickr images first searched by users and then curated into a custom database,

taking advantage of the metadata and effectively cutting down on the post-review of the images. The research of Jiang et al., 2012 and Hodosh et al., 2013 are prominent examples.

3 Methodology

In the previous chapter, we established the primary questions of text auto-illustration and some of the tools used to answer these questions. In this chapter, we present the system that we build for text auto-illustration and its methods of evaluation. We will continue to explain the fundamental technical details to the CV and NLP methods we implement.

As described in Chapter 1, this research was a part of an ongoing research project for vision-impairment tools in INRIA. Given the request of the project, this research on text auto-illustration was intended to supplement short travel texts with images. These constraints motivated the choice to apply a data-correlation and image retrieval approach. An event-depiction approach is better suited for instructional or explanatory texts. Likewise, rendering images was impractical and unnecessary for our purposes. For the choice of images, we want access to a large pool of images and to focus the research on text and image content, rather than on the field of image rendering.

Our approach will focus on illustrating short French-language texts of 200-500 words. We will proceed as follows:

- Create a corpus
- Create queries for image retrieval
- Retrieve images
- Evaluate images and text auto-illustration

Figure 3.1 depicts the pipeline we will create. We start by creating a corpus of documents. For each document, queries are created for image retrieval. Each query is composed of three components that are extracted from the document. Each query is used to retrieve images, each of which are then classified with a label. The label is compared to one of the query components (NER). Based on the results of the comparison, each image is accepted or rejected for illustration.

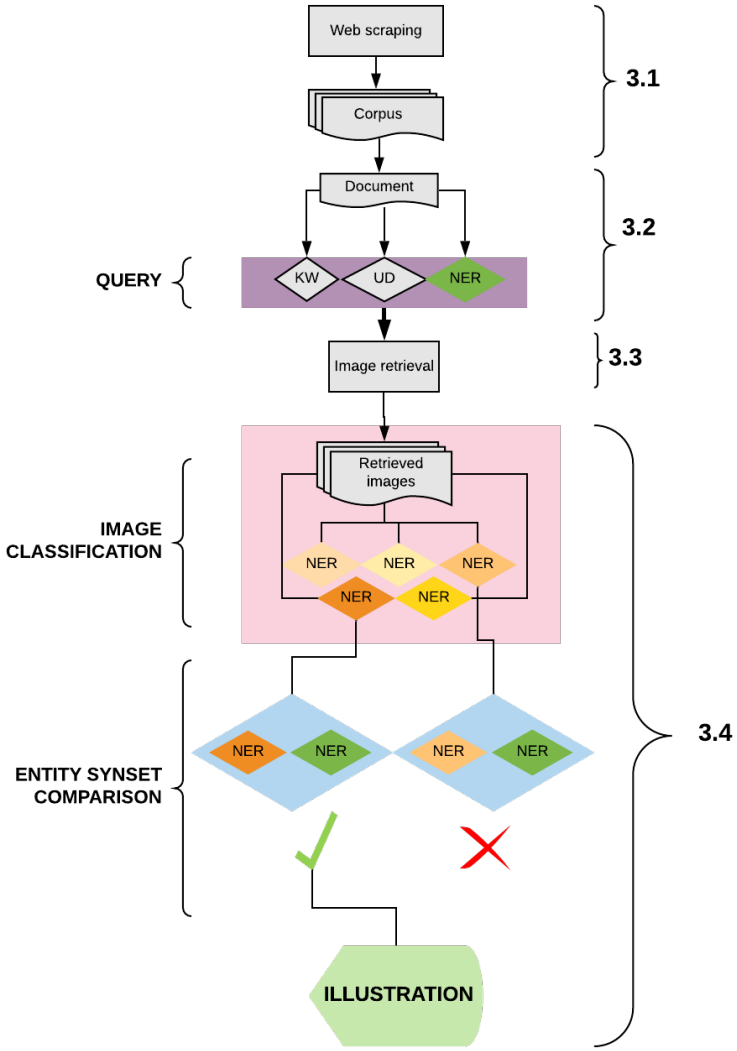

Figure 3.1: Our text auto-illustration pipeline

Our development and test experiments will follow different procedures. In our development experiments, we will ask our human reviewers to evaluate our query components and image analysis models. The texts will be analyzed to determine the best query settings, followed by extensive image evaluation. For our test experiments, our evaluation will be comparatively simple: using the settings motivated by our development test results, we illustrate each text and ask our human reviewers if the illustration is good. Our development experiments are evaluated at each step, whereas only the final result of our test experiment is evaluated.

In Section 3.1, we begin by explaining the corpus. In Section 3.2, we explain the logic of how we build our queries for image retrieval. Next, in Section 3.3, we describe how we retrieve images. Finally, in Section 3.4 we discuss how we analyze our images and how we accept candidates for auto-illustration.

3.1 Corpus building

Creating a corpus is our first step. For the purposes of this research, we collect texts from French travel websites http://routard.com (Routard, 2018) and http://michelin.com (Michelin, 2019). The travel domain was chosen for several reasons. For one, low-vision

patients participating in studies of a reading platform noted that there was a lack of travel resources for low-vision people. There is also precedent for researching the enrichment of travel guides with NLP methods, including image annotation (Fujii et al., 2016), and ontology-based methods (Lam and Lee, 2007). This research precedent supports our domain choice as a conventional one which contributes to the NLP field's scientific research.

Since our second research goal is to compare human evaluation with our automated evaluation, we want our corpus to be small as our human evaluation resources are limited. We decide on a corpus of 200 texts of 200-500 word counts which will be divided into a development and test set with a 30/70 split. To ensure that our development and test sets contain a variety of representative documents, we use the "shuffle" feature in the sci-kit Python library (Pedregosa et al., 2011) using the document word lengths as the relevant feature.

We refer to each scraped travel article as a *document*. Our development set contains 60 documents and our test set contains 140 documents. Both websites are scraped using code written using the BeautifulSoup Python library (Richardson, 2007). The spaCy (Honnibal and Montani, 2017) Python library is used for the text processing. Using spaCy, we establish our base word count by tokenizing each document and considering each alphanumeric token a word.

Each document will be used with different off-the-shelf tools throughout the pipeline that we are building, some working optimally with different types of processed documents. Therefore, we create different version of each document. Using spaCy's lemmatization tools, we create a lemmatized copy of each document. We also add to spaCy's French stop-word list to create a more complete stop-word analysis of each document and to save a version of each document that is free of stop-words. These documents will be used to test the various keyword extraction methods described in Section 3.2.1.

Additional processing accounts for accented characters and documents with compound names. Each document is labeled with a *document name*, which is the title of the scraped article. Some names are compound. In our corpus, which we represent using JSON formatting (Pezoa et al., 2016), we store a key with a *Boolean* value to indicate whether a document name is compound or not. This will help us to understand how many compound names we have in our corpus and to evaluate our keyword methods' success in extracting different types of predicted keywords. Likewise, we store another Boolean to mark if a document name contains an accented letter. Many document names contain accent marks that are native to French writing. For example, words describing something as being related to *Slovénie* (Slovenia) will likely use inflected variations of *slovène*. We store a copy of the document with deaccented letters for the gensim tools (Řehůřek and Sojka, 2010) we use to extract keywords and to similarly account for French accents.

3.2 Query building

In this section, we will explain how we build our queries for each document. First, each document is assigned a keyword. Next, each document is analyzed for named entities. Finally, the dependency relations of these entity relations are analyzed and included or discarded. Each document will have multiple queries, depending on the number of valid entities detected.

query = thematic keyword + named entity + relevant dependencies

Figure 3.2: Query building blocks

3.2.1 Keyword extraction

As described in Section 2.2.3, for our research, every document will have a thematic keyword that will anchor each query. Since we are scraping websites, we have labels of each document in our corpus. We will extract keywords using four methods: TF-IDF, TextRank, RAKE, and LDA. We will then compare the labels of our corpus to the results of each keyword approach to determine which method works best for our corpus. We follow three steps: run the keyword algorithms, evaluate precision recall of the results, and analyze the text similarity of the keyword with the original and processed text.

For our implementation of TF-IDF, we use the TfidfVectorizer from the scikit-learn Python library (Pedregosa et al., 2011). We set the n-gram range to 1-2, allowing us to consider bi-grams in the TF-IDF count. We run TF-IDF on two variants of our stop-word free corpus: one version containing only nouns, and the other with all tokens.

For our implementation of RAKE, we use an open source Python implementation (Medelyan, 2018). We test with variations of minimum character length, maximum word length, and minimum frequency. We use the tokenized and lemmatized version of our corpus, with stop words removed.

For our implementation of TextRank, we use the keywords implementation in the gensim Python library (Řehůřek and Sojka, 2010). We allow bigrams and use the deaccent feature. We run TextRank on two variants of our stop-word free corpus: one version containing only nouns, and the other with all tokens.

For our implementation of LDA, we use the scikit-learn Python library (Pedregosa et al., 2011). We do informal experiments of the LDA algorithm before settling on the parameters described in Figure 3.3. Unlike the other keyword experiments, we allow LDA to consider keywords that range up to five tokens in length. We run LDA on two variants of our stop-word free corpus: one version containing only nouns, and the other with all tokens.

Before we can compare the methods to each other, we tune the keyword approaches to ensure that we were comparing optimized versions of algorithms to each other. In addition to tuning the individual parameters of each method, we also run the keyword algorithms on the original tokenized documents and on tokenized documents in which stop words are removed. In Figure 3.3, we outline the development parameters we use.

- TF-IDF: standard algorithm
- TextRank: allow compound keywords; maximum length of 5 words; remove accentuation
- LDA: maximum 10 iterations; online learning method, learning decay of .8, learning off-set of 50
- RAKE: variations of minimum character length, maximum word length, and minimum frequency (2, 2, 1), (2, 4, 1), (3, 3, 1), (3, 4, 1), (3, 2, 1), (3, 3, 2), (3, 4, 2)

Figure 3.3: Keyword algorithm parameters

To evaluate the results of each of these keyword extraction methods, we define what we consider to be a positive and negative match. In Figure 3.4, we define TP, FP, TN, and FN.

- True Positive (TP): The KW is the exact match of the document label.
- False Positive (FP): The KW result was a non content-bearing result that was not a noun.
- True Negative (TN): 0 keywords found.
- False Negative (FN): A KW result that was a possible choice but not an exact match.

Figure 3.4: True positive explanation

An example of a FP is the identification of the verb *plaire* (to please) as the keyword. An example of a FN is the identification of the adjective *danois* (Danish) for Denmark. We can similarly account for document labels that are compound or for keyword results that have the same sense as our document label. For example, in the Greek island example, if one keyword method found *grecque* (Greek) instead of *îles grecques* (Greek isles) we would still consider *grecque* (Greek) to be a good keyword match.

For the final comparison of the results, we compare the keyword extraction methods using precision, recall, accuracy, and the F-score (Goutte and Gaussier, 2005). The values are calculated with manually written Python code.

Keyword and document similarity

The first step to creating our query is determining a thematic keyword for each query. We assume that the document labels are good keywords, but we are unsure if this label is the most representative keyword for image queries. For example, a "Bordeaux" keyword for an article about Bordeaux may be conventional. However, for articles labeled *Etats unis* (United States) and *îles grecques* (Greek islands), "USA" or "Greece" or specific island names may be used more conventionally within a text and in metadata labeling. However, we want to verify if the computational expense of keyword extraction justifies using an extract keyword over the document label as a thematic keyword. We want to know if there are dramatic differences in the keyword that our methods use and the document names that we assume to be good labels.

The purpose of the thematic keyword is two-fold: it describes the general text and it helps us query a curated resource, Flickr. For this first purpose, we will see if the best keyword or document label is similar to the text. For our similarity measure, we use the cosine similarity that is the default similarity feature in the spaCy library. We write code to iterate through the corpus and use spaCy's similarity measure to make the four comparisons described in Figure 3.5.

- Compare the best keyword to tokenized document without stop words.
- Compare the best keyword to original, tokenized document.
- Compare document label to tokenized document without stop words.
- Compare document label to original, tokenized document.

Figure 3.5: Keyword similarity measures

Keyword and document use in image retrieval

To answer our second query building experiment goal, we want to see if the thematic keyword is helpful on its own to query Flickr. Assuming our thematic keyword is a good match for meta data, we want to verify that it will help us find pictures. We perform a Flickr query with the keyword label and the document label on the development set only. The assumption is that the results on the development set will determine the final test set parameters.

We use the FlickrAPI Python library (Stüvel, 2019) to access the Flickr API. We use the "text" parameter that matches the query to a "free text search" and retrieves photos whose title, description or tags contain the text. For each document in the development set, we record the number of results for two keywords: the best performing keyword and the document label. We want to see if there is a big disparity between the number of image retrievals for the extracted keyword search and the document label search. If there is, this, along with the development results of the similarity results described in Section 3.2.1, will help determine the final choice of the thematic keyword for the test set.

3.2.2 Named entity recognition

For this research, we decide to make named entities the picturable subjects that we aim to illustrate. To this end, we will exploit the semantic information that the BabelNet API provides. To identify the entities we will be picturing, each document will be analyzed with Babelfy. The goal for the named entities is to find which concepts and entities found by Babelfy are most pertinent and most picturable. Babelfy recognizes both abstract concepts and named entities. Abstract concepts are more difficult to picture and can be ambiguous. We assume that such concepts will create noise in our image retrieval; that is, querying for images with abstract concepts will retrieve images that are not relevant to our text. To limit the noisy data that we will retrieve, we want to filter the Babel-detected concepts and entities. To do this initial filtering, we will use the WordNet synsets and taxonomic information, as described in Section 2.2.1, to decide which entities and concepts will be relevant.

We access the Babelfy API through the HTTP API (Navigli, 2020) with Python code. We also use the NLTK Python library (Loper and Bird, 2002) and its Open Multilingual WordNet corpus (Bond and Paik, 2012) to access relevant synset information for the entities detected by Babelfy. Our work consists in iterating through the synset of each identified entity.

For our development NER experiments, we want to find if there exists a common synset path for abstract entities that is picturable. For each identified entity, we proceed by iterating through the synset's full path. All entities share a root *entity*. We accept entities that have a root+1 of *physical_entity.n.01* because they correlate to entities that we assume to be picturable, such as *athletes*, *sand*, *national park*, and *lowlands*.

The bulk of the work is analyzing entities that have *abstraction.n.06* in their paths to see if they are picturable candidates. This category is more difficult to evaluate because of some abstract concepts that we consider more ambiguous and less picturable, such as *warmth* or *tradition*. However, some entities with *abstraction.n.06* as their root+1 are picturable. For example, *Roman architecture*, *waltzes*, *surfing*, and *fishing* all have *abstraction.n.06* as their root+1. These examples we assume to be good candidates for visualization.

For our development experiments, we save the paths for the synset of each identified entity in a JSON file. We accept all entities for image retrieval. We will show all images to our human reviewers, who will evaluate the images as described in Section 3.4.2.

We will compare the reviewers' responses to the synset paths to see if there is a relationship between the synset path and the number of images retrieved and their picturability. Based on these development results, we will determine which named entities we will accept and reject for our test set.

3.2.3 Dependencies

So far, our query is composed of a named entity and a thematic keyword. We also want to account for any of the entity's neighboring words which may improve our query and our chances of retrieving relevant images. For example, if Babelfy identifies *Alps*, we would like to know if the preceding word is *French* or *Italian*. The goal for this analysis is to find any dependency relations that coincide with how users tag images and that will help us improve our named entity-based queries.

- Purpose: Identify relevant dependency relations with the identified entity.
- Experiment goals:
    1. Determine the window for dependency relations that are semantically interesting for creating image queries.
    2. Determine if neighboring entities should be used in separate queries.

Figure 3.6: UD development set experiments

We describe the goals of our dependency analysis in our development in Figure 3.6: first, to determine the size of the "window" in which we search for relevant dependencies; secondly, to determine which dependencies we want to accept.

The analysis is done with a Python program that iterates through the corpus. The spaCy library and its French model are used to label the corpus with dependency labels. The processing of the entities was extensive because of hyphenation, spacing, and some compound named entities being composed of two or more entities within themselves. Some entities may be neighboring. For these cases, we must make a decision whether to combine entities or if we can determine a criteria for keeping entities together or separating them. We iterate through the development set's entities to record these statistics and to present the "windows," including these contingent entities, to our human reviewers. Based on their human evaluation of the entities, we will determine the final settings for the test set.

We set the initial window for the development set to include the three right-hand neighbors. Since some entities are multi-token entities, the "window" for each entity will vary. This decision for the length of the window for the development test is motivated by the characteristics of French syntax. The *X of Y* is a common descriptive construction. We anticipate that relevant descriptive dependency relations will include constructions where X is the entity and Y is a relevant property.

To determine the best dependency relations to include in our queries, we rely on the human reviewers' evaluation of the development set. We present the full "window" of each entity to the reviewer. In their evaluation, the reviewers record the number of dependency relations, if any, are relevant and meaningful. We do not show them the dependency relation. Based on their reviews, we verify if there is a pattern between their evaluation and the dependency labels. Any meaningful patterns will be used to determine the final query.

Final query

As described, our development tests will determine our final query to be used for the test set experiments. We choose our test query based on the optimal features determined by the development experiments. In Figure 3.7, we summarize the experiments on the development set that determine our final test query.

- Keyword: keyword match to document label; number of Flickr results of a keyword candidate; text similarity of keyword to document and document label to document.
- Named entities: accept/reject criteria based on synset paths and human evaluation of picturable entities.
- Additional descriptors: size of window for dependency analysis; human evaluation of best dependencies.

Figure 3.7: Development set experiments

3.3 Image retrieval

In the previous section, we described how the queries related to each text are created: thematic keyword + named entity + its relevant dependents. Each document will contain multiple queries based on the number of relevant entities. Each query will then be used to retrieve images to illustrate the texts. We retrieve the images from Flickr. As described in Section 2.4.2, Flickr has been used in research and publications in both CV and NLP. It also has an active community of users who interact through user groups and conventions that are represented through meta-data, notably through the use of *tags*. To retrieve the images, we use different sets of queries to experiment with our thematic keyword and dependency relations choices and to test if one query retrieves better results than the other. We assume that some queries will give us very few photos, while other queries will retrieve too many photos to sift through. To see what gives us the most consistent number of results, we experiment with some basic search parameters.

Meta-data attached to images in Flickr comes in many forms, such as geo-tags and time stamps, but we were concerned with data containing words. The relevant meta-data for us was the "all" option for the text, which is to say the tags, popularized as hashtags, image captions, and image titles. For all of our searches, we want to make sure our query searched all of these sources. We use our queries to retrieve images that have any metadata containing our search query.

We write Python code to retrieve images using the FlickrAPI Python library (Stüvel, 2019) and to download images. For the development set test, we count the number of images retrieved using the best keyword extracted and the document label. We use this to contextualize the keyword decision as a part of the query development tests.

3.4 Image classification

In this section, we describe how we classify and analyze images, and finally evaluate the image classification. For this final evaluation, we compare two ways of evaluating the images: the image contents using CV methods and human review.

For the image analysis, the Python library Keras (Chollet et al., 2015) is used with architectures using ImageNet models. Our goal will be to see which model will be

best at identifying image contents. With the Babelfy NER, each identified entity has an assigned WordNet synset. Likewise, the ImageNet models classify objects in an image using WordNet synsets. With each query, we will have two synsets: one for the entity which is the root of the query, as identified by Babelfy, and the second, which is identified in the image with the image classifying algorithms. A perfect match is one where the synset of the named entity matches the synset identified in the retrieved image. We use this assumption to automatically illustrate text: images that are good candidates to illustrate a named entity in a text will contain objects that have shared synsets with the named entity in the text.

The synset identified in an image will depend on the results of the computer vision models and architectures that are used. To this end, we will experiment with two different architectures using ImageNet weights to compare results.

3.4.1 CV evaluation

Our image content experiments test CV models' evaluation of our retrieved images. Our overall goal is to determine which computer vision model identified images and their contents: MobileNet or Resenet. For each entity and its query, we retrieve a maximum of 10 photos. We do not retain every photo, but the ones that are most relevant. Each image is analyzed with the ResNet and MobileNet models and is labeled with a WordNet synset. We then compare if this synset corresponds to the synset of the entity that was in the query. We define matches as described in Figure 3.8.

- Match: A perfect match with the entity's synset label and the image's classification.
- Related match: A match with the lowest common hyponym being at least two "steps" from the root synset, *entity*, in its path.

Figure 3.8: Synset matches

3.4.2 Human evaluation

For the human evaluation, a reviewer will judge if a retrieved image matches the entity recognized in the text. We make the distinction that the reader will not judge the WordNet label, but will evaluate if an image is a good conventional representation of the named entity it is supposed to illustrate. This means that the human evaluation will not be of the query and of the whole system, but the evaluation of the images that are retrieved and considered a match according to the MobileNet and ResNet models. The human evaluation will be of the image that the CV model chooses. The reviewer will not evaluate what LDA keyword is best or what UD relationship should have been used or even which CV model is best. Instead, the reviewer will evaluate the image as it relates to the query.

With our reviewers, we want to recreate the process of our automated system. For the automated image evaluation, we input our images into the models described in Section 2.1.3, which classify each image with a WordNet synset. We then take this synset and compare it to the synset of the named entity that was used to retrieve our image. Automatically, we write a Python program to check if the two entities are a match. We want to create the same system with our reviewers: we ask them if the synset of a text's named entity matches the image retrieved using that entity.

We will have two different stages of human review for the development set and the test set. For the development set, we want our human reviewers to help us build the

best model. We ask our reviewers to evaluate the matches according to the standards described in Figure 3.9.

- **True Positive**: the CV recognized entity is a perfect match to the text entity; the reviewer agrees.
- **False Positive**: the CV recognized the image as being a perfect match to the entity; the reviewer disagrees.
- **True Negative**: the CV did not recognize the image as being a perfect match to the entity; the reviewer agrees.
- **False Negative**: the CV did not recognize the image as being a perfect match to the entity; the reviewer disagrees.

Figure 3.9: Questions for human reviewers

With the evaluation of the development set images, we want to be able to answer the questions in Figure 3.4.2

- **Keyword**: Which thematic keyword retrieves the best images: the document label or the LDA keyword?
- **Text entities**: Should we disregard some entities identified in the text as bad candidates for picturability, based on their synset?
- **CV model**: Which CV model performs best: ResNet or MobileNet?
- **Text entity + image matches**: Should we accept images that are not a perfect match? If so, what should the criteria be?

Figure 3.10: Questions to determine settings for test set

Our human evaluation of the test set will be simpler. We will ask our reviewers to review the actual illustration of the texts. Our question will expect a binary answer to the question: is this a good illustration of the text?

4 Results

In this chapter we present our results. In the first part, we present the experiments on our development set. The results of the development set experiments will determine the settings for the test set. To this extent, the presentation of the development set results will also include explanations why some results are informative. In the second part, we present our final results on our test set.

4.1 Corpus

We opted for a 30/70 split of our data set, with our development set being composed of 60 documents, and our test set being composed of 140 documents. Our development set contains less data than our test set, which is unconventional, and is motivated by our human reviewer resources. Our development results will require more manual evaluation than our test set. Each document represents a travel text and is is labelled with the title of the travel article it contains.

4.2 Development experiments

In this section, we report the results of our development experiments. We describe the features analyzed and how they motivate our final features and models for the test set.

4.2.1 Query building

Keyword extraction

The purpose of these experiments was to see if our different keyword-finding methods could find keywords that were most useful when query images for matches with the keyword and the image metadata. We can match against the pages that we scraped.

Our first approach was to look for exact matches. However, we do not expect the key words to necessarily match the document label because there may be documents which describe something specific that is more relevant for image retrieval.

The results of the keyword algorithms were evaluated by humans with the logic explained in Figure 3.4.

Algorithm	Doc	Precision	Recall	F-score	Accuracy
TF-IDF	full	67.80	100	80.80	40.00
	nouns	62.70	100	77.10	37.00
TextRank	full	67.80	100	80.80	40.00
	nouns	58.60	97.10	73.10	34.00
LDA	full	69.50	100	82.00	41.00
	nouns	54.20	100	70.30	32.00
RAKE	222	0	0	0	1.00
	221	8.16	66.70	12.50	4.15
	241	8.90	55.60	15.40	5
	331	19.20	83.30	31.30	10.10
	341	9.10	55.60	15.60	5.00
	321	50	100	66.70	2.00
	332	50	100	66.70	2.00
	342	50	100	66.70	2.00

Table 4.1: Development results for KW

LDA performed the best. Some texts were exceptional. For example, for *Louisiane* (Louisiana), the American state, all of the noun models were better than the original.

For many of the false positive results, both the noun-only and complete corpus created the the same sort of false positive results. For example the document labeled *Pakistan* (Pakistan) had "Lahore," the capital of Pakistan, as its keyword. The same is true for *Russie* (Russia) and *Moscou* (Moscow).

Keyword and document similarity

We found that there was no meaningful difference between the similarity measures. This testing is not exhaustive, but it is an indication that we can query images with both the LDA keyword and with the document label that we have. We are most interested in the results of the LDA keyword in order to be able to build our unsupervised system. However, the results of these similarity measures indicate that our keyword and document label are on par in terms of describing a document. Since the labels are good, we will also experiment with making queries using the document label instead of the keyword. We will use them to make a separate set of queries.

Keyword and document use in image retrieval

As described in Section 3.2.1, we want to make sure that our keywords and document labels are useful in retrieving images from Flickr. In Figure 4.2.1, we record our results.

- **33/60**: Same number of image retrievals for LDA keyword and document label.
- **13/60**: The LDA keyword retrieved more images.
- **14/60**: The document label retrieved more images.

Figure 4.1: Image retrieval on Flickr using the document labels and extracted keywords.

We are most interested in the number of differences. In the cases where the LDA keyword retrieved more images, there was either a large or small difference. This was usually because the LDA keyword was much more general. For example, for the Bahamas, the LDA keyword was *île* (island) with 2,369,430 image retrievals. The

document label was *bahamas* (Bahamas) with 921,391 retrievals. The LDA keyword retrieved 1,448,039 more images, but this is not necessarily desirable for us since many *île* photos may not be relevant to our document about the Bahamas.

4.2.2 Named entity recognition

In the development set, the average word count was 270.5 words, including stop words, per document. Each document had an average of 47.5 entities. Of those entities, an average of 3 entities occurred more than once in a document.

All of the detected entities have a shared root of *entity.n.01*. After this root, there are two major categories after the root: *abstraction.n.06* and *physical_entity.n.01*. When evaluating the detected entities, we divide our workflow into the two categories. The first goal was to determine if there were any categories that could be disregarded. Table 4.2 shows our full results. Note that the table includes double-counts. To evaluate the acceptable roots and paths, we were most concerned not with the unique entities, but with the total occurrences and the number they represented in the total development set.

paths		# in dev set	A/R	summary
physical_entity.n.01	object.n.01	1015	A	geographical objects
	process.n.06	46	A	abstract processes
	causal_agent.n.01	191	A	A individuals and positions
	thing.n.12	110	A	landmarks
	matter.n.03	92	A	natural entities
abstraction.n.06	psychological_feature.n.01	462	A	emotions, expressions
	group.n.01	160	A	organizations and groups
	communication.n.02	117	A	languages, communication
	relation.n.01	96	A	relations between entities
	attribute.n.02	318	A	descriptive words
	measure.n.02	128	A	measures
	set.n.02	3	R	includes units

Table 4.2: WordNet labeling of recognized entities in the development set

4.2.3 Dependencies

As explained in Section 3.2.3, we search for the entity's relevant descriptors for our image queries. Table 4.3 contains the top dependency relations in the development set documents used to make queries, including the percentage of total detected relations they represent. The relations are of adjectival and nominal modifiers.

To answer the first experiment goal, we examine the 1,611 different kind of relations that were found in the development set, as shown in Table 4.3. The results in Table 4.4 hint at what a knowledge of French syntax suggests: most descriptors will either follow a noun or will be connected through a case relationship. Practically that means that our entity is followed by an adjective, such as *villages* or with a prepositional phrase, such as *villages du charme*. With these tests, we found that most windows under 3 were not relevant.

dep. relation	label	# in set	% in set	example
nmod, det	core nominal phrase, function word	78	2.6	*moldavie des*
obl, det	non-core nominal phrase, function word	69	2.3	*écoliers aux*
obj, case	secondary core nominal phrase, function word	47	1.5	*chapelet de*
obj, det	secondary core nominal phrase, function word	45	1.5	*phoques du*
obl, case	non-core nominal phrase, function word	37	1.2	*turquie de*

Table 4.3: Most common UD relations occurring in the development set. The first relation represents the entity.

To answer our second experiment goal, we examine contingent entities or entities in the window. Often this occurred with nouns that have the same token as their inflected adjectival form. For example, in a text about Germany (*Allemagne*), we encountered the phrase *villages de charme* where both *villages* and *charme* were identified as entities with the dependencies *(villages, obl)*, *(de, case)*, and *(charme, nmod)*. *Charme* (charm) was used in its noun form instead of as the adjective *charmant* (charming). This is a stylistic choice which results in Babelfy recognizing two entities. For such cases, we decide to ignore any contingent dependency relation. The reason being that, sometimes, the contingent entities sound semantically more powerful together. However, this is not always the case. Our ultimate goal is to be able to use these core queries and the WordNet label in the previous part to later verify if matched images contain the entity. If we mix entities and put two semantic labels in one query, we will not be able to properly measure if the models trained on ImageNet data recognized one or the other. Even if combining the entities may, on some occasions, create better sounding queries, the added benefit is ambiguous and stands in the way of the ultimate research goal.

Considering these findings and the characteristics of French syntax, we accept all UD relations with the 3 right hand neighbors where the final token has a `amod` or `nmod` dependency label. For the smaller relationships, we cut off at the `amod` or `nmod` since our first analysis of the UD lists showed that the most semantically interesting tokens had these labels. In the development set, we found 307 relevant relations, which are shown in Table 4.4.

rank	dependency relation	label
1	nmod	570
2	obj	512
3	obl	414
4	nsubj	225
5	conj	206
12	obj amod	28
13	nmod amod	27
15	obl amod	19
16	obj case det nmod	18
19	obl case det nmod	17

Table 4.4: Most common UD relations, along with the top 5 most common UD relations that are not single-word entities.

Final development set query

Given the results of our findings on the initial development set query, we decide on the configurations in Figure 4.2 for our development set image retrieval.

- **Keyword**: Try two sets of queries, one using the LDA keyword and one using the document label.
- **Named entities**: Accept all entities as recognized by Babelfy, except for abstractions with *set.n.01* in the root+2 position in the synset path.
- **Additional descriptors**: Accept the first three right-hand neighbors of an entity if the final token has an `amod` or `nmod` dependency label.

Figure 4.2: Development set query settings

4.2.4 Image retrieval

In Table 4.5 we show the results of image retrieval results with the different keyword options. We do not want queries that retrieve too many images that we have to filter through, nor do we want 0 image retrievals. We store the top 10 images returned by the Flickr search, which are returned by relevance.

	query match	0	<10	>10
LDA	partial	826	501	1713
Doc. label	partial	777	508	1755

Table 4.5: Numbers show the number of queries which retrieved 0, less than 10, or more than 10 images.

Both the LDA and document label as keywords had similarly large numbers of large retrievals, but the LDA keyword had more 0-image retrievals. Likewise, the exact matches tended to close up the image search possibilities and resulted in more 0-image results than the partial match search. Based on these results, we decided to perform a partial match search using the document label as our thematic keyword. We collect the top ten results of the query.

4.2.5 Image classification

CV evaluation

Table 4.6 includes the entity roots identified in the retrieved images. Our motivation to count these entity roots is to see if there are some general tendencies of the models and the synsets that they identify. This table can be compared to Table 4.2, which outlines the root paths for the entities recognized by Babelfy. In this table, we list the root paths that the CV models recognizes in the photos. We see that both models on the two different set of images identify more objects with a *physical_entity.n.01* lowest common hyponym in its synset path than with the *abstraction.n.06* lowest common hyponym.

root	LCH	MobileNet		ResNet	
		LDA	Doc. label	LDA	Doc. label
physical_entity.n.01	object.n.01	18408	18898	18488	19015
physical_entity.n.01	matter.n.03	257	277	196	189
abstraction.n.06	communication.n.02	82	100	76	86
physical_entity.n.01	causal_agent.n.01	73	96	75	95
abstraction.n.06	attribute.n.02	39	35	24	21

Table 4.6: The distribution of lowest common hyponyms identified by each CV model with the two query sets.

Our justification for only retrieving 10 images per query is the computational expense and image relevancy. As shown in Table 4.5, each query produced many results. In Figure 3.8, we define what we consider a match and a related match. Table 4.7 shows the results of the synset matches in our retrievals.

match	MobileNet		ResNet	
	LDA	Doc. label	LDA	Doc. label
0 imgs	826	777	826	777
exact match	96	119	104	129
related match	2484	2625	2495	2601

Table 4.7: CV model image classification synset compared to the text entity synsets

Of these matches and related matches, we want to determine if the CV models has difficulty identifying certain entities. The image results could suggest entity filtering that should be applied at earlier stages of our system, namely at the query-building stage.

synset	MobileNet		ResNet	
	LDA	Doc. label	LDA	Doc. label
castle.n.02&	18	23	27	32
restaurant.n.01&	12	14	15	17
cinema.n.02&	2	12	2	12
seashore.n.01&	5	12	4	11
lion.n.01&	10	10	10	10
cliff.n.01 &	7	9	6	9
african_hunting_dog.n.01	8	8	7	7
volcano.n.02	4	7	4	7
palace.n.04&	7	7	6	6
timber_wolf.n.01	5	5	6	6

Table 4.8: Synsets with the most perfect text entity + image matches.

Table 4.8 shows the perfect entity + text matches. We see that, as we suspect given the findings in Table 4.6, the perfect matches are all physical objects.

Human evaluation

The second part of our research goal is to see if the results of the automated text illustration system we built can be compared to a human's evaluation. As described in Section 3.4.2, our human evaluation of the development will review if the images

retrieved match our query. To this end, we want to compare our reviewer's evaluations to the evaluations of the CV models.

We considered showing our reviewers all the retrieved images for the 60 documents in our development set, as shown in Table 4.5, but the number of images exceeded our resources for practical human evaluation. Likewise, we are mostly interested in the false positives and false negatives that the models detect so that we can present the best possible auto-illustrations as part of our test set evaluation.

To this extent, instead of showing our human reviewers all the images retrieved, we show them all the images retrieved for all the queries in the documents that had at least one "perfect" match as shown in Table 4.7. Since many of the documents had many queries that received 0 results, we hope that this would help us to better see what was and was not working for the image retrieval and evaluation. Some documents, such as the text for *Géorgie* (Georgia), had only 1 perfect match, but had image retrievals for partial matches, so we were able to have a sizable amount of partial match images for human review.

Since we do some initial filtering of our images, we skew our results because we ignore partial matches with the lowest common hyponym being less than two "steps" away from the root synset, *entity.n.01*. We effectively ignore a pool of potential false negatives; in our tests this amounts to 16,265 images with the LDA-ResNet analysis; 16,284 images with LDA-MobileNet; 16,676 images with document label and ResNet; and 16,662 images with the document label and MobileNet analysis.

In Table 4.9, we see our reviewer's results. Our definition of a true positive (TP), false positive (FP), true negative (TN), and false negative (FN) is presented in Section 3.4.2.

	MobileNet		ResNet	
	LDA	Doc. label	LDA	Doc. label
TP	93	118	104	129
FP	3	1	0	0
TN	439	289	442	290
FN	780	1056	769	1045
Total imgs	1315	1464	1315	1464
Accuracy	41%	28%	42%	29%
F-1 score	19%	18%	21%	2%

Table 4.9: The human evaluation of the images retrieved for the 23 documents with at least one perfect match

With this feedback in Table 4.9, we want to answer the questions asked in Section 3.4.2.

- **Keyword**: The document label keyword retrieved more images that ended up having TPs than the LDA keyword. The high accuracy for the LDA keyword queries suggest that the LDA keyword is better, but this high accuracy is due to the difference in the false negatives and true negatives, which was a choice of the entity paths that we decided to accept. To decide on which keyword is best, we are more concerned with the true positives because questions of which synsets to accept or not does not confuse our evaluation of which keyword is best. To that extent, the document label performed better than the LDA keyword.

- **CV model**: To determine the best configuration between the ResNet and MobileNet models, we are interested in the true negatives and the true positives.

We can see that the ResNet model performs slightly better at identifying perfect matches and discarding bad matches.

For the remaining questions about the relevant text entities and the acceptable synset matches to accept (as described in Section 3.4.2), we are interested in reducing the number of false negatives. We want to capture patterns that will help us refine our entity filtering. Our initial experiment had a good rate of true positive and false positives, but many potentially good images were lost as false negatives. We are interested in finding patterns in the common paths of synsets that are lost as false negatives, but we want to avoid over-fitting. We iterate through the different lowest common hyponyms collected and record patterns. In Table 4.6, we show the most common entity roots. In this analysis, we look for the specific synsets within the false negatives. First, in Table 4.10, we show the top true negatives for the retrieved images which the reviewer deemed that the CV models correctly rejected. We can consider eliminating these in our test set since they are attached to images that we know to be true negatives.

paths	MobileNet		ResNet	
	LDA	Doc. label	LDA	Doc. label
artifact.n.01	169	108	178	114
...	98	71	93	66
...artifact.n.01, structure.n.01	24	25	26	18
...artifact.n.01, instrumentality.no3	7	14	11	9
...living_thing.n.01, organism.n.01, animal.n.01	0	11	1	11

Table 4.10: Top lowest common hyponym for true negative results. All paths have the root path entity.n.01, physical_entity.n.01, object.n.01, whole.n.02

Next, in Table 4.11, we look at the top paths of the most common false negatives. There is overlap with the top 5 most common synset paths of the false negatives and true negatives, so the first intuition to exclude the true negative results would also exclude false negatives. However, aside from these top 5, the false negative paths were longer than the true negative results. This suggests that both models, MobileNet and ResNet, have difficulty making an exact match for objects that aligns with the text entity, but they come very close. For example, each model with each query had more than 20 false negatives of *bird.n.01* and *vertebrate.n.01*, related to *animal.n.01*. Instead of looking for common roots, it may be easier to eliminate false negatives that have a lowest common hyponym that is of a similar length to the recognized text entity.

path	MobileNet		ResNet	
	LDA	Doc. label	LDA	Doc. label
whole.n.02, artifact.n.01	179	215	180	236
geological_formation.n.0	121	178	105	171
whole.n.02	131	165	128	155
whole.n.02, artifact.n.01, structure.n.01	58	77	61	79
whole.n.02, artifact.n.01, instrumentality.n.03	15	30	14	27
# of unique paths	**44**	**59**	**39**	**54**

Table 4.11: Lowest common hyponym for false negative results. All paths have the root path entity.n.01, physical_entity.n.01, object.n.01...

- **Text entities**: The false negatives all had mostly objects, which were also the most well recognized true positives. Abstract terms were not on the list. No obvious subset of entities was suggested to be eliminated.
- **Text entity + image matches**: We can see that, for all models, there is an overwhelming number of false negatives. We accept perfect matches and images whose classification synset has a lowest common hyponym that is no more than two paths shorter than the text entity's synset.

4.3 Final results on the test set

The parameters of our analysis on the 140 documents of our test set were based on the results reported in Chapter 4. Based on these development experiments, our final round of text-illustration was done with the settings presented in Figure 4.3.

- **Query**: Document label + all entities + 3 right hand tokens if they have an *amod* or *nmod* dependency relation.
- **Image retrieval**: A partial match in all of the image's meta data.
- **Image evaluation**: The ResNet model. Accept images with perfect matches, as well as those whose classification synset has a lowest common hyponym that is no more than two paths shorter than the text entity's synset.
- **Human evaluation**: A binary yes/no approval of a text's illustration with retrieved images.

Figure 4.3: Settings for test set experiments

4.3.1 Query building

The test set is composed of 120 documents. We keep the same UD relations as in the development set. We also retrieve with all of the entities recognized, opting to do the filtering on the image retrieval part. Table 4.12 lists the most common synset in the recognized entities in the test set.

root+1	root+2	# of entities
physical_entity.n.01	object.n.01	3083
abstraction.n.06	psychological_feature.n.01	1168
abstraction.n.06	attribute.n.02	930
physical_entity.n.01	causal_agent.n.01	479
abstraction.n.06	group.n.01	470

Table 4.12: Top 5 synset roots in identified entities.

4.3.2 Image analysis

Table 4.13 lists the perfect matches of the test set, along with the related matches with the updates consideration of the lowest common hyponyms. Of the 140 documents in the test set, 55 contain perfect matches.

	Entities
Total text entities	7645
0-images	1950
perfect match	287
related images	390

Table 4.13: Entities in test set.

Human evaluation

The human evaluation for our test set was different from our task in the development stage. For the development, we wanted to understand what impact our parameters for the query building and for the image retrieval and the different models had on the photos that were identified as matches. For our test set, we want to evaluate the configurations that we have chosen based on the results of our development tests. For this human evaluation, we only judge the photos that are chosen to illustrate text. Of the 140 test documents, 55 had at least one perfect text entity. We choose these documents for our test set review.

Of the 55 documents, all but 4 documents were judged as having "good" illustrations. Two documents contained perfect matches, while the other two included partial matches. Examples of accepted and rejected images in the test set are included in Appendix 5.1.

5 Conclusion

In this research, we have examined how NLP and CV resources can be united to facilitate research in text auto-illustration, a topic that involves both textual analysis and the treatment of images. Our original goals were to:

- Build an unsupervised system for text auto-illustration for short (200-500 words) French texts.
- Compare human evaluation of text auto-illustrations with the automated evaluation by CV models.

For our first goal, we succeeded in building a system that can be practically adapted to short texts in other domains. Although our research focused on French texts, we made use of multi-lingual resources, namely the WordNet ontology, BabelNet, and Babelfy, and our image retrieval used Flickr, a website with meta-data in users' languages. We built our own corpus of web-scraped documents, for which the document label was the only labeled information that we used for our experiments. For our second goal, we gave human reviewers a standard against which to judge the results of our text auto-illustration pipeline. We used the human evaluation of our development set to decide on our final settings for our test set. Our human reviewers then evaluated the auto-illustrations that our automated system evaluated as being the best matches. We found that our human reviewers agreed with our automated system's illustrations.

This automated system has two parts: the textual analysis and image analysis. Our textual analysis is performed with the goal to create a query for image extraction. Our query is composed of a recognized entity, dependency relations, and a thematic keyword. Our initial emphasis on keyword extraction was motivated by our goal to build a system that was not dependent on labeled data or domain-specific resources.

To this end, we started with experimenting with different keyword extraction methods. We then performed two rounds of similarity experiments: we compared our extracted keyword, using the LDA algorithm, and our document label to the original text. The majority of our extracted keywords were identical to our document labels, while the similarity of both to our original texts was comparable. We ultimately used the document labels as our thematic keyword in order to optimize our results, but the development experiments are promising for further work that has unlabeled data.

Likewise, the dependency relations that we included in the query were based on manual evaluation of which of an identified entity's neighboring words were meaningful. This step can be adapted for other languages, but the multilingual labeling system of Universal Dependencies make the identified relations, `amod` and `nmod`, reasonable choices for other languages.

The image analysis part of our system produced simple but satisfying results. We compared the results of two different CV architectures, MobileNet and ResNet, which classify images with a WordNet synset. Our goal was to compare each image's classification with the text entity's synset. This was the main innovation of this research and allows us to automate text auto-illustration. Our initial concern with the recognized entities was the potential to create queries with abstract entities with low picturability. However, the ImageNet data set has physical entities over represented. Abstract entities ended up not being pictured. Instead, the system retrieved many

more false negatives that could have been included in the texts' illustration. This was our system's weakness, but it did not prevent us from reaching our second goal of comparing the CV evaluation to a human reviewer's. We allowed our model to accept image entities that had a high lowest common hyponym with the identified text entity. These pictures, along with the perfect matches, were deemed to be good images for text auto-illustration. Appendix 5.1 contains examples of retrieved images.

5.1 Future work

Our results suggest that there is promise in building text auto-illustration systems that are not dependent on labeled data or curated resources. However, low recall of our image evaluation prevented a full auto-illustration of texts. Our system had good results with identifying true positives, but many retrieved images and entities were not illustrated. Future work could focus on how to deal with these false negatives. To deal with this in our development experiments, we used human input on our results and "climbing" WordNet synset paths to find the best lowest common hyponym. Additional work could include using a corpus of picturable synsets or picturable tokens in order to explicitly include the concept of picturability into our system. Another option is to use a semantic model with machine learning tools to identify potential entities, which are then later linked to a WordNet synset that can be correlated with ImageNet data.

Future work could also focus on segmenting the short texts for better presentation. A topic modelling algorithm, LDA, provided the best keyword results, but future work could also apply topic modelling to the text. Instead of one thematic keyword, a document could be segmented into separate parts based on topics. Each part could have its own thematic keyword. This could result in retrieving better images and in reducing redundant retrievals.

Our work also ignores polarity, as we depended on Flickr's safety filters and censorship policies. However, including polarity in our textual analysis could help filter out undesirable terms for illustration. Including some filter during the image analysis is also possible. Likewise, we do not account for the possible bias that images can trigger. For example, in a document labeled "Botswana," the entity *embouchure* (delta) was pictured with an illustration from a colonial perspective. Such images can introduce a bias for readers which future research should strive and work to avoid.

Appendix: Text auto-illustration

Accepted and rejected test documents

Examples of accepted and rejected illustrations.

Accepted image: perfect text+entity synset match

Figure 5.1: Text entity: `volcano.n.02`
*...des terres modelées par la force tellurique des **volcans**.*
Translation: ...land formed by the force of **volcanoes**.

Accepted image: partial text+entity synset match

Figure 5.2: Text entity: `whale.n.02`
*...des excursions en bateau pour aller saluer les **baleines**...*
Translation: ...boat excursions for **whale** watching...

Rejected image: partial text+entity synset match

Figure 5.3: Text entity: `table.n.02`
*Enfin, vous trouverez aussi plutôt facilement une bonne **table** aux tarifs raisonnables à Brest.*
Translation: You will easily find a variety of reasonable **dining options** in Brest.

Rejected image: perfect text+entity synset match

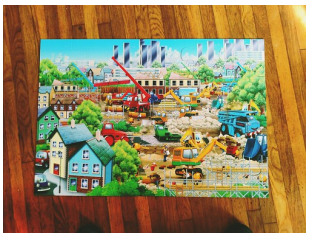

Figure 5.4: Text entity: `jigsaw_puzzle.n.01`
*...voilà une agglomération sans centre véritable, sorte d'immense **puzzle** de 88 quartiers.*
Translation: ...an agglomeration with a real center, making up a **puzzle** of 88 neighborhoods.

Example of keyword aid

An example of the same entity, *la côte* (the seashore, *seashore.n.01)*, retrieving different results with different thematic keywords.

Figure 5.5: Text entity: `seashore.n.01`
*...la Costa Brava est le nom touristique donné à la **côte** catalane...*
Translation: ..Costa Brava is the touristic name given to the Catalan **coast**...

Figure 5.6: Text entity: `seashore.n.01`
*...des **côtes** désertiques où la vie semble impossible.*
Translation: Desert-like **coasts** where life seems impossible.

Example of biases in images

Figure 5.7: Text entity: `mouthpiece.n.06`
*Pas d'**embouchure** maritime : le fleuve se déverse dans le désert même.*
Translation: The river flows not into a **delta**, but into the desert itself.

Example of confused model

Examples of false negatives where the text's ambiguity caused a bad perfect match.

Figure 5.8: Text entity: `cellar.n.03`
*Les villes de la Slovaquie comptent de nombreuses cours et **caves** chargées d'histoire à découvrir par soi-même...*
Translation: Slovakia's cities have numerous yards and **bars** full of history to discover...

Examples of a false negative where the text's ambiguity caused a mismatch between the text entity and image entity.

Figure 5.9: Text entity: `bow_tie.n.01`
*Les parcs nationaux et les réserves naturelles, farouchement gardés, abritent la plupart des espèces animales d'Afrique australe (36 de grands mammifères, 600 d'oiseaux, 80 de poissons, 70 de serpents, 500 de **papillons** et 3 000 de plantes)...*
Translation: The national parks and natural reserves, heavily guarded, contain the majority of African species (36 of large mammals, 600 of birds, 80 of fish, 70 of snakes, 500 of **butterflies**, and 3,000 of plants)...

Example of similes that work

Figure 5.10: Text entity: `kaleidoscope.n.02`
*Montréal est une ville comme un **kaléidoscope**, enrichie par un étonnant mariage de communautés.*
Translation: Montreal is like a **kaleidoscope**, enriched by a stunning marriage of communities.

Bibliography

Agrawal, Rakesh, Sreenivas Gollapudi, Anitha Kannan, and Krishnaram Kenthapadi (2011). "Enriching textbooks with images". In: *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, pp. 1847–1856.

Alm, Norman, Mamoru Iwabuchi, Peter N Andreasen, and Kenryu Nakamura (2002). "A multi-lingual augmentative communication system". In: *ERCIM Workshop on User Interfaces for All*. Springer, pp. 398–408.

Barnard, Kobus (2016). "Computational Methods for Integrating Vision and Language". *Synthesis Lectures on Computer Vision* 6.1, pp. 1–227.

Baroni, Marco, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta (2009). "The WaCky wide web: a collection of very large linguistically processed web-crawled corpora". *Language resources and evaluation* 43.3, pp. 209–226.

Blei, David M, Andrew Y Ng, and Michael I Jordan (2003). "Latent dirichlet allocation". *Journal of machine Learning research* 3.Jan, pp. 993–1022.

Bond, Francis and Kyonghee Paik (2012). "A survey of wordnets and their licenses". *Small* 8.4, p. 5.

Cao, Liangliang and Li Fei-Fei (2007). "Spatially coherent latent topic model for concurrent segmentation and classification of objects and scenes". In: *2007 IEEE 11th International Conference on Computer Vision*. IEEE, pp. 1–8.

Carney, Russell N and Joel R Levin (2002). "Pictorial illustrations still improve students' learning from text". *Educational psychology review* 14.1, pp. 5–26.

Chollet, François et al. (2015). *Keras*. https://keras.io.

Coyne, Bob and Richard Sproat (2001). "WordsEye: an automatic text-to-scene conversion system". In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, pp. 487–496.

Coyne, Bob, Richard Sproat, and Julia Hirschberg (2010). "Spatial relations in text-to-scene conversion". In: *Computational Models of Spatial Language Interpretation, Workshop at Spatial Cognition*.

Del Pero, Luca, Philip Lee, James Magahern, Emily Hartley, Kobus Barnard, Ping Wang, Atul Kanaujia, and Niels Haering (2011). "Fusing object detection and region appearance for image-text alignment". In: *Proceedings of the 19th ACM international conference on Multimedia*, pp. 1113–1116.

Delgado, Diogo, Joao Magalhaes, and Nuno Correia (2010). "Assisted news reading with automated illustration". In: *Proceedings of the 18th ACM international conference on Multimedia*. ACM, pp. 1647–1650.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Dwivedi, Priya (2019). *Understanding and Coding a ResNet in Keras*. URL: https://towardsdatascience.com/understanding-and-coding-a-resnet-in-keras-446d7ff84d33 (visited on 2019-01-06).

Ferraresi, Adriano, Silvia Bernardini, Giovanni Picci, and Marco Baroni (2010). "Web corpora for bilingual lexicography: A pilot study of English/French collocation extraction and translation". *Using Corpora in Contrastive and Translation Studies. Newcastle: Cambridge Scholars Publishing*, pp. 337–362.

Fillmore, Charles J and Collin Baker (2010). "A frames approach to semantic analysis". In: *The Oxford handbook of linguistic analysis*.

Firoozeh, Nazanin, Adeline Nazarenko, Fabrice Alizon, and Béatrice Daille (2020). "Keyword extraction: Issues and methods". *Natural Language Engineering* 26.3, pp. 259–291.

Flickr (2019). *Flickr*. URL: http://flickr.com (visited on 2019-08-03).

Fujii, Kazuki, Hidetsugu Nanba, Toshiyuki Takezawa, and Aya Ishino (2016). "Enriching travel guidebooks with travel blog entries and archives of answered questions". In: *Information and Communication Technologies in Tourism 2016*. Springer, pp. 157–171.

Goutte, Cyril and Éric Gaussier (2005). "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation". In: *ECIR*.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hernandez, Nicolas, Fabien Poulard, Matthieu Vernier, and Jérôme Rocheteau (2010). "Building a French-speaking community around UIMA, gathering research, education and industrial partners, mainly in Natural Language Processing and Speech Recognizing domains". In:

Hodosh, Micah, Peter Young, and Julia Hockenmaier (2013). "Framing image description as a ranking task: Data, models and evaluation metrics". *Journal of Artificial Intelligence Research* 47, pp. 853–899.

Honnibal, Matthew and Ines Montani (2017). "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing". *To appear*.

Howard, Andrew G, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam (2017). "Mobilenets: Efficient convolutional neural networks for mobile vision applications". *arXiv preprint arXiv:1704.04861*.

Huang, Chieh-Jen, Cheng-Te Li, and Man-Kwan Shan (2013). "Vizstory: Visualization of digital narrative for fairy tales". In: *Technologies and Applications of Artificial Intelligence (TAAI), 2013 Conference on*. IEEE, pp. 67–72.

ImageCLEF (2019). *ImageCLEF*. URL: https://www.imageclef.org (visited on 2019-08-03).

INRIA (2019). *INRIA*. URL: https://www.inria.fr/en/ (visited on 2019-08-30).

Itabashi, Y and Y Masunaga (2005). "Correlating scenes as series of document sentences with images-taking the tale of genji as an example". In: *Proc. of IEEE International Conference on Data Engineering Workshops (ICDE'05)*, p. 1235.

Jiang, Yu, Jing Liu, Zechao Li, Changsheng Xu, and Hanqing Lu (2012). "Chat with illustration: a chat system with visual aids". In: *Proceedings of the 4th International Conference on Internet Multimedia Computing and Service*. ACM, pp. 96–99.

Johansson, Richard, Anders Berglund, Magnus Danielsson, and Pierre Nugues (2005). "Automatic text-to-scene conversion in the traffic accident domain". In: *IJCAI*. Vol. 5, pp. 1073–1078.

Jones, Karen Sparck (1972). "A statistical interpretation of term specificity and its application in retrieval". *Journal of documentation*.

Jurafsky, Daniel and James H Martin (2000). "Chapter 8: Word classes and Part-Of-Speech Tagging". *Speech and Language Processing, Prentice Hall*.

Kelman, Avi, Michal Sofka, and Charles V Stewart (2007). "Keypoint descriptors for matching across multiple image modalities and non-linear intensity variations". In: *2007 IEEE conference on computer vision and pattern recognition*. IEEE, pp. 1–7.

Kesorn, Kraisak, Sutasinee Chimlek, Stefan Poslad, and Punpiti Piamsa-Nga (2011). "Visual content representation using semantically similar visual words". *Expert Systems with Applications* 38.9, pp. 11472–11481.

Krawczak, Karolina, Małgorzata Fabiszak, and Martin Hilpert (2016). "A corpus-based, cross-linguistic approach to mental predicates and their complementation: Performativity and descriptivity vis-à-vis boundedness and picturability". *Folia Linguistica* 50.2, pp. 475–506.

Kunda, Maithilee (2018). "Visual mental imagery: A view from artificial intelligence". *Cortex* 105, pp. 155–172.

Lam, Toby HW and Raymond ST Lee (2007). "iJADE FreeWalker–An Intelligent Ontology Agent-based Tourist Guiding System". In: *Computational Intelligence for Agent-based Systems*. Springer, pp. 103–125.

Lau, C, Dian Tjondronegoro, Jinglan Zhang, Shlomo Geva, and Yue Liu (2006). "Fusing visual and textual retrieval techniques to effectively search large collections of wikipedia images". In: *International Workshop of the Initiative for the Evaluation of XML Retrieval*. Springer, pp. 345–357.

Learning, Deep (2019). *Deep Learning Implementations*. URL: https://www.deep-learning-site.com/implementations.html (visited on 2019-08-07).

Li, Haojie, Jinhui Tang, Guangda Li, and Tat-Seng Chua (2008). "Word2image: towards visual interpreting of words". In: *Proceedings of the 16th ACM international conference on Multimedia*. ACM, pp. 813–816.

Loper, Edward and Steven Bird (2002). "NLTK: The Natural Language Toolkit". In: *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics*.

Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". *International journal of computer vision* 60.2, pp. 91–110.

Ma, H., J. Zhu, M. R. Lyu, and I. King (2010). "Bridging the Semantic Gap Between Image Contents and Tags". *IEEE Transactions on Multimedia* 12.5 (Aug. 2010), pp. 462–473. ISSN: 1520-9210.

Ma, Hao, Jianke Zhu, Michael Rung-Tsong Lyu, and Irwin King (2010). "Bridging the semantic gap between image contents and tags". *IEEE Transactions on Multimedia* 12.5, pp. 462–473.

May, Wesley, Sanja Fidler, Afsaneh Fazly, Sven Dickinson, and Suzanne Stevenson (2012). "Unsupervised disambiguation of image captions". In: ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pp. 85–89.

McDonald, Ryan, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. (2013). "Universal dependency annotation for multilingual parsing". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 92–97.

Medelyan, Alyona (2018). *A python implementation of the Rapid Automatic Keyword Extraction*.

Medhi, Indrani, Aman Sagar, and Kentaro Toyama (2006). "Text-free user interfaces for illiterate and semi-literate users". In: *2006 International Conference on Information and Communication Technologies and Development*. IEEE, pp. 72–82.

Michelin (2019). *Michelin Travel Guides*. URL: http://michelin.com (visited on 2019-08-07).

Mihalcea, Rada and Chee Wee Leong (2008). "Toward communicating simple sentences using pictorial representations". *Machine translation* 22.3, pp. 153–173.

Mihalcea, Rada and Paul Tarau (2004). "Textrank: Bringing order into text". In: *Proceedings of the 2004 conference on empirical methods in natural language processing*.

Miller, George A, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller (1990). "Introduction to WordNet: An on-line lexical database". *International journal of lexicography* 3.4, pp. 235–244.

Moro, Andrea, Alessandro Raganato, and Roberto Navigli (2014). "Entity linking meets word sense disambiguation: a unified approach". *Transactions of the Association for Computational Linguistics* 2, pp. 231–244.

Navigli, Roberto (2020). *Babelfy API*. URL: http://babelfy.org/guide (visited on 2010-12-06).

Navigli, Roberto and Simone Paolo Ponzetto (2012). "BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network". *Artificial Intelligence* 193, pp. 217–250.

Norvig, P Russel and S Artificial Intelligence (2002). *A modern approach*. Prentice Hall.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). "Scikit-learn: Machine Learning in Python ". *Journal of Machine Learning Research* 12, pp. 2825–2830.

Pezoa, Felipe, Juan L Reutter, Fernando Suarez, Martın Ugarte, and Domagoj Vrgoč (2016). "Foundations of JSON schema". In: *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, pp. 263–273.

Qin, Danfeng, Christian Wengert, and Luc Van Gool (2013). "Query adaptive similarity for large scale object retrieval". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1610–1617.

Řehůřek, Radim and Petr Sojka (2010). "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, May 2010, pp. 45–50.

Richardson, Leonard (2007). "Beautiful soup documentation". *April*.

Rose, Stuart, Dave Engel, Nick Cramer, and Wendy Cowley (2010). "Automatic keyword extraction from individual documents". *Text mining: applications and theory*, pp. 1–20.

Routard (2018). *Routard*. URL: https://www.routard.com/ (visited on 2018-08-02).

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei (2015). "ImageNet Large Scale Visual Recognition Challenge". *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

Santini, Simone (2001). "Mixed media search in image databases using text and visual similarity". In: *ICME 2001, IEEE International Conference on Multimedia and Expo*. Citeseer.

Sarma, Himangshu, Robert Porzel, Jan D Smeddinck, Rainer Malaka, and Arun Baran Samaddar (2018). "A Text to Animation System for Physical Exercises". *The Computer Journal* 61.11, pp. 1589–1604.

Serrano, Navid, Andreas E Savakis, and Jiebo Luo (2004). "Improved scene classification using efficient low-level features and semantic cues". *Pattern Recognition* 37.9, pp. 1773–1784.

Stüvel, Sybren A. (2019). *Python Flickr API implementation*. URL: https://github.com/sybrenstuvel/flickrapi (visited on 2010-12-06).

Sudderth, Erik B, Antonio Torralba, William T Freeman, and Alan S Willsky (2005). "Learning hierarchical models of scenes, objects, and parts". In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. IEEE, pp. 1331–1338.

Unsplash (2019). *Unsplash*. URL: https://unsplash.com (visited on 2019-08-03).
UzZaman, Naushad, Jeffrey P Bigham, and James F Allen (2011). "Multimodal summarization of complex sentences". In: *Proceedings of the 16th international conference on Intelligent user interfaces*. ACM, pp. 43–52.
Westerveld, Thijs (2000). "Image retrieval: Content versus context". In: *Content-Based Multimedia Information Access-Volume 1*. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, pp. 276–284.
Wikipedia (2018a). *Relief (disambiguation)*. URL: https://en.wikipedia.org/wiki/Relief_(disambiguation) (visited on 2018-08-02).
Wikipedia (2018b). *Wikipedia*. URL: https://en.wikipedia.org (visited on 2018-08-02).
Yu, Licheng, Eunbyung Park, Alexander C Berg, and Tamara L Berg (2015). "Visual madlibs: Fill in the blank description generation and question answering". In: *Proceedings of the ieee international conference on computer vision*, pp. 2461–2469.
Zhu, Xiaojin, Andrew B Goldberg, Mohamed Eldawy, Charles R Dyer, and Bradley Strock (2007). "A text-to-picture synthesis system for augmenting communication". In: *AAAI*. Vol. 7, pp. 1590–1595.
Zitnick, C Lawrence, Devi Parikh, and Lucy Vanderwende (2013). "Learning the visual interpretation of sentences". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1681–1688.