Prague University of Economics and Business
Faculty of Informatics and Statistics

Aspect-based sentiment analysis of conference review forms

MASTER THESIS

Study program: Applied Informatics
Field of study: Knowledge and Web Technologies

Author: Bc. Sára Juranková
Supervisor: prof. Ing. Vojtěch Svátek, Dr.

Prague, December 2020

Prohlášení

Prohlašuji, že jsem diplomovou práci „Aspect-based sentiment analysis of conference review forms“ vypracovala samostatně za použití v práci uvedených pramenů a literatury.

5.12.2020 .....

Bc. Sára Juranková

Acknowledgements

I would like to thank the thesis' supervisor, prof. Ing. Vojtěch Svátek, Dr., for all his guidance in writing this thesis.

I would also like to acknowledge the suffering my sister had to go through in helping me write this thesis. Few would do what she has done for me.

Abstract

The aim of this thesis is to create a system for extracting opinions and sentiment from conference paper reviews and group these opinions by the different criteria a paper is judged on. The theoretical part of the thesis describes the existing methods of sentiment analysis and natural language processing, thus providing necessary context. The reviewing process of conferences focused on semantic technology and the structure of the reviews are explored. A set of criteria is identified, based on the fields of different conference review forms and used as a foundation for the extraction of terms that are used to express these criteria. A sentiment lexicon is created specifically for the domain of conference paper reviews. In the practical section a dictionary-based sentiment lexicon analysis method is implemented and applied to a set of reviews from 3 different conferences. The results are then evaluated by comparing the numerical scores estimated by the algorithm with the numerical scores from the reviews. The outcome is then explored further, by inspecting the accuracy of criterion identification and sentiment analysis on a sentence level. The precision of criterion identification is evaluated at 57.38 % and the recall at 53.44 %, while the sentiment polarity is correct in over 75 % of cases. The rationale behind this outcome is explained and a set of recommendations is given for future improvements.

Keywords

sentiment analysis, conference submission reviews, aspect-based sentiment analysis

Abstrakt

Cílem této práce je vytvoření systému pro extrakci názorů a sentimentu z recenzí konferenčních příspěvků a seskupování těchto názoru podle kritérií, na základě kterých jsou tyto příspěvky posuzovány pro akceptaci. Teoretická část práce uvádí existující metody analýzy sentimentu a zpracování přirozeného jazyka. Následně prozkoumává strukturu recenzí z konferencí zaměřených na sémantické technologie a proces, kterým tyto recenze vznikají. Na základě struktury recenzních formulářů z různých konferencí je navržena obecná množina kritérií, která jsou v této práci vytvářeným systémem z recenzních textů extrahována. Ta pak slouží jako báze k extrakci výrazů, které je vyjadřují. Rovněž je vytvořen lexikon slov se sentimentovou polaritou, specifický pro konferenční recenze. Tento lexikon je následně využit pro implementaci metody analýzy sentimentu. Ta je následně aplikována na množinu recenzí ze tří různých konferencí. Výsledky numerických odhadů pro jednotlivá kritéria jsou porovnávány s vlastním číselným hodnocením autorů recenzí. Výstup systému je dále zkoumán na úrovni vět pro zjištění správnosti identifikace kritérií a polarity sentimentu. Výsledná přesnost implementovaného algoritmu při identifikaci kritérií vychází na 57.38 % a úplnost na 53.44 %, přičemž úspěšnost klasifikace sentimentu činí zhruba 75 %. Dosažené výsledky jsou zhodnoceny a jsou navržena doporučení pro budoucí zlepšení systému.

Klíčová slova

analýza sentimentu, recenze konferenčních příspěvků, aspektová analýza sentimentu

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

EKAW European Knowledge Acquisition Workshop
ESWC European Semantic Web Conference
FN False Negative
FP False Positive
ISWC International Semantic Web Conference
MAE Mean Absolute Error
ML Machine Learning
NLP Natural Language Processing
NLTK Natural Language Toolkit
POS Part Of Speech
ST Semantic Technology
TP True Positive

Introduction

The aim of this thesis is to create a system for extracting opinions and sentiment from conference paper reviews. In other words, the goal is to build a system which will allow to automatically determine the opinion of the author of the review on different aspects of the reviewed paper. Papers submitted to a conference go through a reviewing process, the goal of which is to decide whether the paper will be accepted to the conference or not. The structure of these reviews varies across conferences as well as individual reviewers. Sometimes comments in the reviews are separated by the different criteria the paper is judged by (such as presentation or relevance to the conference). Other times they are separated by the positive and negative remarks, while some conferences require the reviewers to give numerical scores to a set of chosen criteria. However, often there is no structure required.

Because each submitted paper is reviewed by many people, and because each conference has its own structure of the review form as well as a different set of criteria it is difficult to quickly get an idea of the quality of the submitted work. Being able to get a general idea of how good a paper is as well as swiftly assessing what are the strong and weak parts of a submission is especially important for meta-reviewers during the discussion periods, as their goal is to provide a summary of the more in-depth reviews.

Therefore, the idea behind this thesis is to create a process able to determine these qualities and assess the value of the reviewed paper in order to improve the mechanics of acceptance or rejection.

Extracting the opinions of the reviewer on a paper, mapping it onto a unified set of criteria and transforming it into a numerical value could significantly simplify the process of submission acceptance. It can also provide a way to compare reviews of the same paper across different conferences and reviewers.

The ability to extract opinions of reviewers on different criteria could also provide a way to carry out a larger study to ascertain which aspects of a paper get criticized routinely and why. This has a potential to help future authors to improve their papers accordingly before submission and therefore reduce the risk of their paper being rejected by avoiding the common mistakes of other authors.

My motivation for the topic of this thesis is that I find the field of sentiment analysis incredibly interesting and I see an immense potential in its application in various domains. So far it has been mostly studied in connection to social networks or product reviews but I believe that it could be also applied to a number of different tasks, such as the one explored in this thesis or for example to study the objectivity of media. The system created here could also serve as a base for a larger review management system that is able to generate a visual metaphor of each review that reflects different review metrics. Visual images are faster and easier to understand than written text, therefore it should make the meta-analysis more

comfortable and effective. The prototype of this visual metaphor generator was already created and accepted as a demo by the International Semantic Web Conference 2020 [1]. Given the fact I took part in this project and would like to see it expand further, the creation of a tool that would allow unstructured reviews to utilize this mechanism is an additional motivation. Especially because at this point the visual metaphor generator only works for reviews containing numerical scores.

In order to implement such a system, it is necessary to create a set of review metrics to which the fields of different conference review forms would be mapped on and which would serve as the criteria extracted from the reviews. Then in order to recognize which of these criteria a reviewer is focusing on in a specific comment a dictionary of terms or aspect expressions, which are linked to these criteria, needs to be assembled. To find out whether a comment on some aspect of the paper is positive or negative a domain-specific sentiment lexicon needs to be compiled. This lexicon consists of words which point to a positive or negative polarity of an opinion. Then a method of aspect-based sentiment analysis needs to be designed and implemented. This technique of information extraction should then be applied to get the opinion of the reviewer on each of these metrics.

The theoretical part of this thesis consists of the following parts: The first chapter introduces the field of sentiment analysis and explains some common tools and methods used to extract sentiment. Tasks and tools of natural language processing are defined and explained in the second chapter, because sentiment analysis is a sub-field of natural language processing. The third and last chapter of the theoretical part offers an insight into the studied domain of conference submission reviews and examines existing research with a similar focus.

The practical part of the thesis is described in chapters four through nine. First the methods chosen for the implementation are described. Chapter four familiarizes the reader with reasoning behind the approach to give them an overarching understanding of the sentiment analysis algorithm. Each step of the implementation is then explained in more detail. The review data were gathered from multiple sources and different preprocessing steps were needed for different tasks of the implementation, therefore the following chapter explains how the data was gathered and prepared for various usages. Chapter six covers the varied methods used to extract aspect or criterion expressions from the reviews. Chapter seven recounts the creation of a sentiment lexicon. The implementation of an aspect-based sentiment analysis algorithm which uses the dictionary of criterion expressions and the sentiment lexicon to discover the reviewer's opinion on the different aspects of the paper is described in chapter eight. Finally section nine provides a complex evaluation of the accuracy of the algorithm based on the estimated numerical scores for the set of criteria. The correctness of the output on a sentence level is also determined in the last chapter of the practical part of this thesis.

1. Introduction to sentiment analysis and existing methods

This chapter serves as an introduction to the field of sentiment analysis and gives an overview of existing tools and current practices.

1.1 Sentiment analysis

Sentiment analysis (also known as opinion mining) is a type of text analysis focused on detecting polarity (e.g. positive or negative opinion) within text. This section aims to explain the basics of the field.

1.1.1 Levels of sentiment analysis

Depending on the granularity of the sentiment analysis, it can be carried out at different levels [2]:

- **Document level** – determines the sentiment based on the entire text, which is most useful when the document expresses opinion on a single entity.
- **Sentence level** – classifies each sentence as positive, negative or neutral, mostly used for subjectivity classification.
- **Entity and aspect level** – recognizes the different entities described in the text and their aspects and extracts opinions linked with these aspects.

The review forms express different opinions on different aspects of a paper (such as presentation, technical quality or significance of the work), and an opinion on a single one of these aspects can span across many sentences, therefore sentiment analysis should be done on an aspect level.

1.1.2 Definition of opinion

In order to explain the task of opinion analysis, first it is necessary to have a definition of an opinion:

> An opinion is a quintuple,
>
> (e, a, s, h, t)
>
> where e is the opinion (or sentiment) target entity, a is the aspect of said entity, s is the sentiment about the target, h is the opinion holder and t is the time when the opinion was expressed. [3, p. 463]

In the case of conference submission reviews, the *target entity* is the review, the *aspects* are the criteria, the *opinion holder* is the author and the *time* could be for example the review version. For the needs of sentiment analysis in the task of extracting opinions from the final versions of paper reviews, a sufficient definition should be just a tuple (e, s) as there is only one target entity (the review) and there should be only a single opinion holder (the author). As reviews sometimes may have more versions than the final one (for example the reviewer adds comments after a rebuttal), and because sometimes, a review also may contain an opinion of the author of the reviewed paper (a sentence such as *"Although the author believes that his idea is novel, that is not the case"* expresses the opinions of two opinion holders – the author and the reviewer), in future the system may be improved to account for this and therefore use the original quadruple definition of an opinion.

1.1.3 Sentiment analysis tasks

Another thing needing definition is the general way sentiment analysis is done. The task of aspect-based sentiment analysis can be described as a process of six steps, working with [2]:

1. Entity extraction and categorization
2. Opinion holder extraction and categorization
3. Aspect extraction and categorization
4. Time extraction and standardization
5. Aspect sentiment classification
6. Opinion quintuple generation

Since conference paper review only describe one entity (the paper) and all opinions should belong to a single person (the reviewer), those parts of the analysis can be left out, as well as time extraction. Therefore the steps actually taken in the sentiment analysis of conference paper reviews are these:

1. **Aspect extraction and categorization** – extract aspects expressions of the defined review criteria and cluster them based on the criteria they represent.

2. **Aspect sentiment classification** – determine whether an opinion on an aspect is positive, negative or neutral and assign it a numeric value describing the polarity of the opinion as well as the strength of the sentiment.
3. **Opinion tuple generation** – produce the tuples (aspect, sentiment) based on the previous steps.

1.2 Sentiment analysis techniques

There are many techniques of sentiment analysis, some rely on more traditional statistical or machine learning (ML) methods, while some are more rooted in linguistics. An overview of the main approaches to sentiment analysis and aspect detection is pictured in Figure 1.1. This section introduces some popular methods of sentiment analysis.

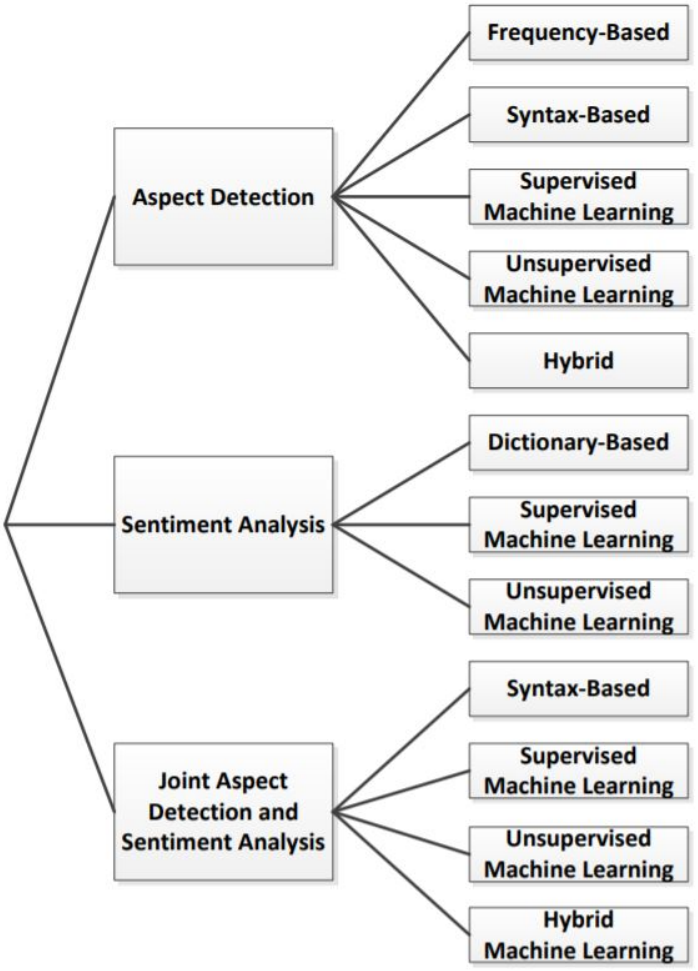

Figure 1.1: Taxonomy for aspect-level sentiment analysis [4].

1.2.1 Machine learning

Machine learning methods, both supervised and unsupervised, can be used for sentiment analysis [2]. For example aspect extraction can be done by Conditional Random Field (CRF), a supervised learning method commonly used in natural language processing [4]. Another technique, which is further explained in section 1.2.2 uses association mining to extract features [5]. Recently there has been a shift towards deep learning methods in various natural language processing tasks, including aspect-based sentiment analysis, with promising results [6, 7]. However, most machine learning methods require high amounts of labeled training data which makes it difficult to transfer a trained model to another domain. Although there are domain adaptation methods, they are primarily focused on sentiment analysis on the document level [2].

One ML technique, which is fairly simple, but often used for simple sentiment analysis is the Naïve Bayes classifier.

The Naïve Bayes classifier

Naïve Bayes is a simple machine learning algorithm that utilizes the Bayes rule together with a strong (or naïve) assumption that the evidence is conditionally independent, given the hypothesis. [8]

Therefore, the equation for the probability of a hypothesis H given a set of evidence $E_1, \ldots, E_K$ that the Naïve Bayes classifier is based on is:

$$P\left(H|E_1, \ldots, E_K\right) = \frac{P(H)}{P(E_1, \ldots, E_K)} \times \prod_{k=1}^{K} P\left(E_k|H\right) \tag{1.1}$$

The goal of the classifier is to determine which of the possible hypotheses is the most probable given the evidence.

In the task of sentiment analysis, the different "classes" that serve as hypotheses are the different sentiment polarities we try to detect. So we can for example have three different classes – positive, negative and neutral.

As was already mentioned, when classifying using the Naïve Bayes algorithm, look for the hypothesis for which the probability is the highest given the evidence. Therefore the equation for $P\left(H|E_1, \ldots, E_K\right)$ can be simplified by leaving the denominator out, because $P(E_1, \ldots, E_K)$ will always stay the same for all possible classes/hypotheses:

$$P(H|E_1, \ldots, E_K) = P(H) \times \prod_{k=1}^{K} P(E_k|H) \tag{1.2}$$

The evidence are the words contained in the text we want to classify. The text is represented as a bag of words, meaning instead of considering the position of each word in the text, we represent it as an unordered set.

When training the Naïve Bayes classifier, the frequency of each word in each text is considered. Each text should also be annotated with its sentiment. The idea behind the bag-of-words approach is depicted in Figure 1.2 .

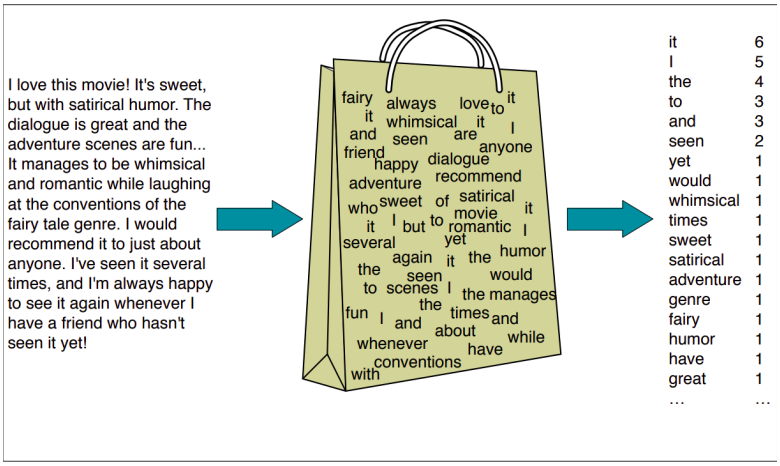

Figure 1.2: Intuition of the multinomial naive Bayes classifier applied to a movie review [9]

Then all the necessary probabilities needed for classification of new texts ($P(c)$ and $P(w_i|c)$ or $P(H)$ and $P(E_k|H)$ from the original formula) are calculated.

The formula for the prior probability of a given class c is:

$$P(c) = \frac{N_c}{N} \tag{1.3}$$

where N_c is the number of text belonging to the class c and N is the total amount of texts in the training dataset.

The probability of a word w_i occurring in a text with a given class c is:

$$P(w_i|c) = \frac{count(w_i,c)}{\sum_{w \in V} count(w,c)} \tag{1.4}$$

where V is the vocabulary, consisting of all words found across all texts, $count(w_i,c)$ is the number of times the word w_i appears in documents belonging to class c and the sum of $count(w,c)$ over all words in the vocabulary calculates the sum of all words in all documents of class c.

Because the Naïve Bayes classifier multiplies all evidence likelihoods together, this equation is usually adjusted to account for the fact that some words might never appear in the training set or never appear in conjunction with some class. This makes their probability given this

class zero and therefore the probability of said class also zero, independently on all the other words that appeared in the text. This behavior is usually corrected by giving these words non-zero probabilities e.g. using the add-one (Laplace) smoothing [9]:

$$P(w_i|c) = \frac{count(w_i,c) + 1}{(\sum_{w \in V} count(w,c)) + |V|} \tag{1.5}$$

To give a clear example of the application of the Naïve Bayes classifier on the task of sentiment analysis, here is how the probability of a sentence *"This phone is great."* being positive would be calculated:

$$\begin{aligned} P(positive|\text{“}this\text{”},\text{“}phone\text{”},\text{“}is\text{”},\text{“}great\text{”}) = \\ = P(\text{“}this\text{”}|positive) \\ \times P(\text{“}phone\text{”}|positive) \\ \times P(\text{“}is\text{”}|positive) \\ \times P(\text{“}great\text{”}|positive) \\ \times P(positive) \end{aligned} \tag{1.6}$$

1.2.2 Dictionary-based approaches

The biggest indicators of sentiment in a text are sentiment words (also called opinion words). These words, often adjectives or adverbs, help to detect the expression of sentiment as well as its polarity. For example, words such as *great*, *amazing* or *good* indicate a positive sentiment, on the other hand words like *terrible*, *awful* or *bad* express negative feelings. [2]

In order to obtain a sentiment lexicon there are 3 main approaches [2]:

- **Manual approach** – is usually combined with automated methods because of its labor intensity.
- **Dictionary-based approach** – usually uses a list of a small number of sentiment words as a seed and then generates the dictionary through tools such as WordNet [10] by integrating their synonyms or antonyms.
- **Corpus-based approach** – the aim is to create a sentiment lexicon for a specific domain, for example by using a seed of sentiment words and then including other words in the lexicon by searching the sentences for conjoined adjectives, where one is already known as a sentiment word.

There are already compiled dictionaries of sentiment words, such as SentiWordNet[1], which assigns to each word both a positive score and a negative score (on a scale from 0 to 1) and allows to obtain an objectivity score based on the two. SentiWords[2] or SenticNet[3] assign to

[1] `https://github.com/aesuli/sentiwordnet`
[2] `https://hlt-nlp.fbk.eu/technologies/sentiwords`
[3] `https://sentic.net/`

each word they contain a sentiment score between -1 (extremely negative) and +1 (extremely positive).

Lexicon-based approaches of sentiment analysis, as the name suggests, utilize lexicons of sentiment words as well as other constructs like sentiment shifters, but-clauses (sentences containing "but" like words such as *"In general I liked it but there are some issues"*) and other words or phrases that affect sentiment.

Sentimentr

One tool that utilizes a lexicon-based method for sentiment analysis is sentimentr.

As explained in the sentimentr documentation *"sentimentr attempts to take into account valence shifters (i.e., negators, amplifiers (intensifiers), de-amplifiers (downtoners), and adversative conjunctions) while maintaining speed. Simply put, sentimentr is an augmented dictionary lookup."* [11].

The way sentimentr works is that instead of simply comparing the words in a sentence with the sentiment lexicon and judging the sentiment of a sentence based on, say, the sum of polarities of sentiment words found within it, it also adjusts the polarity of each sentiment word based on sentiment shifters found in the proximity of that word. The influence of different valence shifters is as follows:

Negators. Negators are words such as *no*, *not* or *never*. The influence of negators on the polarity of a sentiment word is simple – if the number of negators in the left and right context of a sentiment word is even the polarity stays the same, however if the number is odd, the polarity is negated (so a word with originally negative polarity becomes positive and vice versa).

Amplifiers. Amplifiers are words like *especially*, *major* and *significantly*. As the name suggests they increase (amplify) the polarity of a sentiment word. There is however one exception to this rule – if the number of negators in the context of a sentiment words is odd, the influence of amplifiers is negated, in other words they start working as the de-amplifiers described below.

De-amplifiers De-amplifiers, like *slightly*, *somewhat sort of* etc. work analogously to amplifiers, except they decrease the polarity of a sentiment word instead of increasing it.

Adversative conjunction Adversative conjunctions are perhaps the most complex of the valence shifters. These are words such as *but*, *however* or *albeit*. The relative position of an adversative conjunction to the sentiment word plays an important role when determining

its influence. If the conjunction comes before the sentiment word it increases its polarity, however if it comes after it decreases it. According to the sentimentr documentation *"This corresponds to the belief that an adversative conjunction makes the next clause of greater values while lowering the value placed on the prior clause."*. [11]

The final sentiment score of a sentence is calculated as follows:

$$\text{sentiment}(s) = \frac{\sum_{w_i \in Pol} sentiment(w_i)}{\sqrt{|w_i|}} \tag{1.7}$$

where s is the sentence, w_i is the i-th word of the sentence, Pol is a set of polar/sentiment words in the sentence, $sentiment(w_i)$ is the calculated sentiment of w_i based on the valence shifters and $|w_i|$ is the length of the sentence.

A Holistic Lexicon-Based Approach

Another lexicon-based approach is one called a *holistic lexicon-based approach* [12]. This one, contrary to the *sentimentr* method described above which determines the sentiment on a sentence level, focuses on aspect-based sentiment analysis.

The basic algorithm finds all words or phrases describing features in a sentence as well as opinion (or sentiment) words. Then, for each feature in the sentence, its sentiment score is calculated using the polarity of the opinion words and their distance in the sentence from the feature expression using the following function:

$$\text{score}(f) = \frac{\sum_{w_i : w_i \in s \wedge w_i \in V} w_i.SO}{dis(w_i, f)} \tag{1.8}$$

where:

- w_i is an opinion word
- V is the set of all opinion words
- s is the sentence that contains the feature f
- $dis(w_i, f)$ is the distance between feature f and opinion word w_i in the sentence s
- $w_i.SO$ is the semantic orientation of the word w_i

Then *"If the final score is positive, then the opinion on the feature in the sentence s is positive. If the final score is negative, then the opinion on the feature is negative. It is neutral otherwise."* [12, p. 5].

The algorithm is also extended to deal with negation (by negating the polarity of a sentiment word which follows after a negation word), but-clauses (by first trying to determine the sentiment of an opinion word within the but-clause using the basic algorithm and if the

sentiment score is zero it assigns the negation of the clause before *but*). Then it has these three rules for dealing with context-dependent opinion words:

- **Intra-sentence conjunction rule** – this rule is based on the idea that *"a sentence only expresses one opinion orientation unless there is a 'but' word which changes the direction."* [12, p. 6]. Therefore if the orientation of one opinion word depends on the context, but there is another opinion word in the sentence for which the orientation is known and the clauses containing the two opinion words are connected by a conjunction such as *and*, we can assign that orientation to the context-dependent orientation word as well.
- **Pseudo intra-sentence conjunction rule** – this rule applies to sentences without an explicit conjunction, but otherwise works similarly to the previous rule
- **Inter-sentence conjunction rule** – if the opinion orientation is still undetermined after the application of previous rules, the inter-sentence conjunction rule helps assign the orientation using the context of the surrounding sentences. According to the authors *"The idea is that people usually express the same opinion (positive or negative) across sentences unless there is an indication of opinion change using words such as 'but' and 'however'."* [12, p. 6].

1.3 Aspect extraction

A significant part of a system performing an aspect-based sentiment analysis, is the identification of words that are used to express the different aspects. This section introduces different approaches to the extraction of these aspect expressions.

1.3.1 Frequency-based extraction

In order to extract aspect expressions, it was proposed using a Part-Of-Speech (POS – for a more detailed explanation see section 2.1.3) tagger to identify nouns and noun phrases as possible aspect expression candidates. Then to use a simplified Apriori algorithm to calculate frequency of these candidates and finally keep only the ones that are frequent enough. The reasoning is that the vocabulary people use when commenting on an entity converges so the frequently co-occurring sets of terms should represent the important aspects. [5]

1.3.2 Taxonomy based extraction

Another approach uses the user's prior knowledge of the domain to build a hierarchy of features. The technique uses the previously mentioned unsupervised learning approach based on term frequency and adds user-defined features and similarity matching. This is done in order to eliminate redundancy and to generate a set of features in an organized way, reflecting hierarchical relationships between them. The way the method works is that first a set of *crude*

features is generated using the frequency-based method. Then the WordNet lexical database is used to perform similarity matching between the user-defined taxonomy of features and the crude features. Only the crude features that are similar enough to the features already included in the taxonomy are to be further used. Finally, the newly discovered feature candidates, which passed the similarity matching are inspected by a user and if the user considers these candidates valid feature expression, they are added to the final taxonomy. [13]

1.3.3 Patterns for aspect extraction

A very different technique for aspect extraction is an opinion mining framework that consists of a set of heuristic patterns for extraction of aspects as well as sentiments or opinions. In order to get the pairs of aspects and the related sentiments, the reviews first go through a preprocessing phase, part of which is POS tagging. After that, the POS tagged sentences are matched against a set of patterns. Each pattern is a sequence of POS tags, so in order to get a match against one of them, the sentence must contain that sequence of tags as well. The part of the pattern which is a noun, noun phrase or a verb is then considered to be a candidate term for an aspect while mostly adjectives and adverbs represent the expression of sentiment. [14]

2. Natural language processing

Natural language is any language that has not been artificially constructed but rather has evolved naturally through use and is *"acquired by its users without special instructions as a normal part of the process of maturization and socialization"*. [15, p. 29]. These languages such as English, Arabic, Vietnamese or Hindi differ from non-natural languages, which are constructed for specific uses, like for example programming languages or symbolic languages used for studying logic.

Natural language processing (NLP) is *"an interdisciplinary domain which is concerned with understanding natural languages as well as using them to enable human–computer interaction"* [16, p. 1]. The field of applications of NLP is far reaching and includes use cases such as:

- Identifying spam e-mails [17]
- Chatbots for customer support and engagement [18]
- Improvement in clinical documentation [19]

and many others. An overview of the usage of NLP can be seen in Table 2.1.

Table 2.1: Breakdown of various NLP tasks performed by modern NLP software [18]

Word Tagging	**Sentence Parsing**	**Text Classification**	**Text Generation**
Word segmentation	Constituency parsing	Sentiment analysis	Language modeling
Shallow syntax-chunking	Semantic parsing	Text classification	Machine translation
Named entity recognition	Dependency parsing	Temporal processing	Simplification
Part-of-speech tagging	-	Coreference resolution	Summarization
Semantic role labeling	-	-	Dialogue
Word sense disambiguation	-	-	Question answering

It is also a very hard task, given the fact that natural languages do not adhere to the strict rules non-natural languages usually do.

> *"Human language is highly ambiguous... It is also ever changing and evolving. People are great at producing language and understanding language, and are capable of expressing, perceiving, and interpreting very elaborate and nuanced meanings. At the same time, while we humans are great users of language, we are also very poor at formally understanding and describing the rules that govern language."* [20, p. 1]

In fact, the *imitation game*, a test developed by Alan Turing in the 50's as a test of artificial intelligence, relies on the ability of a computer program to impersonate a human in a written conversation. [21]

Because of its potential, NLP has been studied for decades (the first proposals for machine-translation pre-date the invention of the digital computer [22]).

This chapter serves as an introduction to the field of natural language processing, first describing the common tasks of NLP and then introducing some frequently used tools and methods.

2.1 Common tasks in natural language preprocessing

With many machine learning and data mining tasks, we work with data in a form of a large table, where each row represents one object and each column represents property of the objects. Most ML methods are therefore designed to work with these tables.

When it comes to texts in natural language, they are in their raw form just a series of characters. In order to add some structure to this unstructured data, we usually perform a number of preprocessing steps, which allow us to transform the text into a representation better suited to our needs. This section focuses on the most common preprocessing steps of natural language processing.

2.1.1 Tokenization

The goal of tokenization is to separate the text into smaller units and it is *"a fundamental step in both traditional NLP methods... and advanced deep learning-based architectures"* [23]. The text may be segmented into paragraphs, but most commonly tokenization refers to splitting the text into sentences and words. Tokenization can remove punctuation too, but that may cause issues such as wrongly splitting up abbreviations with periods (e.g., dr.), where the period following that abbreviation should be considered as part of the same token and not be removed. [17]

There are different techniques of tokenization, their usage depends on the language and the purpose of tokenization, so this section provides examples of some commonly used tokenizers.

White space tokenization

White space tokenization splits the text into tokens based on white spaces, such as tabulation characters, spaces, newline characters etc.

Although this is a fast and easy way to implement tokenization, this technique only works in languages where meaningful units are separated by spaces e.g. English, but even then, it does not work well for open compound words such as *living room* or *full moon*. [24]

Dictionary based tokenization

Dictionary based tokenization uses a dictionary of tokens to segment the text. If the token is not found in the dictionary, special rules are applied to tokenize it. [25]

For languages without spaces between words, there is an additional step of word segmentation where we find sequences of characters that have a certain meaning. [24]

Regular expression tokenization

Regular expression tokenizers are rule based tokenizers, which use regular expressions to control the tokenization of text into tokens. [25] It is a useful technique when you want more control over the tokenization of the text by creating your own regular expression by which the text is tokenized.

2.1.2 Stop words removal

Some words, which often appear in analyzed documents, have little informational value and can be excluded. This process is called stop words removal. Stop words removal includes getting rid of common language articles, pronouns and prepositions such as *and*, *the* or *to* in English and other words that may be considered insignificant. [17]

Generally, the more often a word or a term appears in a collection of documents, the lesser informational value it has. Therefore the strategy for creating a list of stop words is to sort the terms by the total number of times each term appears in the document collection and pick the most frequent terms as stop words. [26]

Stop words removal is an especially important step in the field of information retrieval, where excluding words with little informational value tends to have a huge impact on the speed of these systems as well as the volume of data that has to be stored.

2.1.3 Part-of-speech tagging

Part-of-speech tagging (POS) is the process of assigning part-of-speech tags to words in a sentence. A POS tag is a label assigned to each token in a document to indicate the part of speech and often also other grammatical categories such as tense or number (plural/singular). [27]

Because POS taggers usually tend to take into account different grammatical categories, their tag set tends to be larger that the number of part-of-speech categories of the language they are intended for. In English, there are frequently listed and thought eight parts of speech (noun, pronoun, verb, adjective, adverb, preposition, conjunction, article and interjection),

but the commonly used Penn Treebank tagset contains 36 POS tags and 12 other tags (for punctuation and currency symbols). [28]

2.1.4 Lemmatization and stemming

Since documents use different forms of a word, we often need to reduce the inflectional forms and sometimes the derivationally related forms of a word to a common base form. This is especially important (and difficult) with synthetic languages which can be defined as *"any language in which syntactic relations within sentences are expressed by inflection (the change in the form of a word that indicates distinctions of tense, person, gender, number, mood, voice, and case) or by agglutination (word formation by means of morpheme, or word unit, clustering). Latin is an example of an inflected language; Hungarian and Finnish are examples of agglutinative languages."* [29].

The two techniques of text normalization are stemming and lemmatization.

Stemming

The algorithms knows as stemmers produce *stems* of a word by cutting off the beginning and the end of the word, usually by using a list of common prefixes and suffixes. [30]

It is a crude heuristic process, which is why the produced stems often do not correspond to the morphological root of the word. If given the token *saw*, stemming might return *saw* (or possibly just *s*, whereas lemmatization (described in the next section) would likely return either *see* or *saw* depending on whether the use of the token was as a verb or a noun. [31]

Lemmatization

Lemmatization is a process of applying morphological analysis to words in order to remove inflectional endings and transform the words into their base or dictionary forms called *lemmas*. Lemmas unlike stems are actual language words. [26]

In lemmatization the normalization depends on the part of speech of a word so it either has the be automatically determined in the previous step or this context has to be supplied in another way. Lemmatization is more sophisticated than stemming, producing more accurate results and meaningful tokens by considering the context, however it has is trade-offs. Compared to stemming, the process of lemmatization is slower and significantly harder to implement.

2.2 Selected (Python-based) NLP tools

As Python is *"the leading coding language for NLP because of its simple syntax, structure, and rich text processing tool"* [32] it has been chosen as the programming language for the implementation of the aspect-based sentiment analysis. This section provides a brief introduction to two of Python's NLP libraries – NLTK and spaCy.

2.2.1 Python's NLTK library

Python's Natural Language Toolkit (NLTK) is an open source library which contains a wide range of tools and algorithms for building programs aimed at natural language processing. It provides a way to perform standard NLP tasks such as part-of-speech tagging, syntactic parsing or text classification. An overview of some NLTK modules with their functionalities is depicted in Table 2.2.

Table 2.2: Language processing tasks and corresponding NLTK modules with examples of functionality [33]

Language processing task	**NLTK modules**	**Functionality**
Accessing corpora	nltk.corpus	Standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	Tokenizers, sentence tokenizers, stemmers
Part-of-speech tagging	nltk.tag	n-gram, backoff, Brill, HMM, TnT
Classification	nltk.classify, nltk.cluster	Decision tree, maximum entropy, naive Bayes, k-means
Chunking	nltk.chunk	Regular expression, n-gram, named entity
Parsing	nltk.parse	Chart, feature-based, unification, probabilistic, dependency
Evaluation metrics	nltk.metrics	Precision, recall, agreement coefficients
Applications	nltk.app, nltk.chat	Graphical concordancer, parsers, WordNet browser, chatbots

WordNet interface

NLTK also provides an interface to WordNet, which is a semantically oriented dictionary of English consisting of 155289 words and 117659 synonym sets [34].

The WordNet interface allow us to find a list synonyms to a given word, which in the context of WordNet is called a *synset*. In order to get a synset the `synsets()` function needs to be called with the word for which the synsets are wanted as an argument along with an optional part-of-speech tag:

```
from nltk.corpus import wordnet
syns = wordnet.synsets("dog")
print(syns)
```

The function outputs a list of synsets:

```
[Synset('dog.n.01'), Synset('frump.n.01'),
 Synset('dog.n.03'), Synset('cad.n.01'),
 Synset('frank.n.02'), Synset('pawl.n.01'),
 Synset('andiron.n.01'), Synset('chase.v.01')]
```

Each synset is identified with a 3-part name in the form <lemma>.<pos>.<number>, where:

- **<lemma>** is the lemma of the word
- **<pos>** is a part-of-speech tag
- **<number>** is the sense number used to disambiguate word meanings [35]

The interface also implements a number of ways how to find similarity between two synsets. Here are some examples along with their description taken from the WordNet interface howto guide [36]:

- `path_similarity` returns a score denoting how similar two word senses are, based on the shortest path that connects the senses in the is-a (hypernym/hyponym) taxonomy.
- `lch_similarity` returns a score denoting how similar two word senses are, based on the shortest path that connects the senses (as above) and the maximum depth of the taxonomy in which the senses occur.
- `wup_similarity` returns a score denoting how similar two word senses are, based on the depth of the two senses in the taxonomy and that of their Least Common Subsumer (most specific ancestor node).

Another function WordNet offers is the `derivationally_related_forms()` function which allows us to find terms which are in a different syntactic category but have the same root form and are semantically related to a word we supply as an argument.

2.2.2 spaCy

spaCy is NLP library for Python and Cython (a programming language written mostly in Python with additional C-inspired syntax). It is a newer library than NLTK, using an object-oriented approach as opposed to NLTK's string approach. Unlike NLTK it has support for word vectors (multi-dimensional meaning representations of a word) and its processing is generally faster than that of NLTK (due to its Cython implementation). Same as NLTK it provides tools for many NLP tasks such as POS tagging, tokenization, measuring similarity between words etc. [37]

3. Description of the domain of analyzed conference paper reviews

The data on which the analysis will be done are all from events and conferences focused on semantic technology (ST). Sentiment analysis is usually applied to product reviews or posts on social media, therefore this work will serve as an exploration of the possibility of creating a model which, even though focused on one domain, is generalized across various events of the same umbrella subject.

Eventually, the model can be extended to be less domain-specific, given that the review metrics probably will not differ significantly between different research areas (or at least not if we stay in the field of technology related conferences).

In this chapter a brief introduction is given into the studied domain of conferences with focus on semantic technology and the general reviewing process a submission goes through. It is also focused on existing research on the topic of conference paper reviews.

3.1 Studied conferences within the field of semantic technology

This section serves as a description of the conferences from which the reviews studied in this work came from, which are generally focused on the topics of semantic technology and knowledge engineering.

3.1.1 European Semantic Web Conference ESWC

The European Semantic Web Conference (ESWC) is an international conference on the topic of ST which first began in 2004. According to the ESWC website, *"the mission of the ESWC is to bring together researchers and practitioners in all these areas dealing with different aspects of semantics on the Web."* [38].

The topics of ESWC conferences include linked open data, machine learning, natural language processing and information retrieval, ontologies, reasoning, semantic data management, services, processes, and cloud computing, social Web and Web science, in-use and industrial, digital libraries and cultural heritage, and e-government. [39]

3.1.2 European Knowledge Acquisition Workshop EKAW

The European Knowledge Acquisition Workshop (EKAW) started in 1987 as a workshop in the field of knowledge-based systems and became a conference in 2000 changing its full title

to the International Conference on Knowledge Engineering and Knowledge Management. [40]

As they state on their website about the 2020 EKAW conference *"the 22nd International Conference on Knowledge Engineering and Knowledge Management is concerned with all aspects about eliciting, acquiring, modeling and managing knowledge, and the construction of knowledge-intensive systems and services for the semantic web, knowledge management, e-business, natural language processing, intelligent information integration, and so on"* [41].

3.1.3 International Semantic Web Conference ISWC

The International Semantic Web Conference (ISWC) focuses on research regarding semantic web topics including linked data. It is a successor of the Semantic Web Working Symposium (SWWS) and is held annually since 2002. [42]

3.2 Reviewing process of conference submissions

The aim of this section is to give an overview of the steps of a reviewing process, to give the reader a better understanding on the context in which the reviews analyzed in this work are created. While the reviewing process of different conferences may vary, there is set of general steps which they all follow to a certain degree.[1]

Each paper submitted to a conference is reviewed by multiple reviewers (usually three or more). After the reviewers submit their reviews authors may be able to react during a rebuttal phase, by clearing up any misunderstandings, answering questions posed by the reviewers and by defending their position on things a reviewer may have complained about. This is a common reviewing step in bigger conferences such as ISWC or ESWC however that may not be the case for smaller conferences such as EKAW where the authors only get the final reviews and whether the submission was accepted or not. Sometimes the authors get the numerical scores given by the reviewers as well as the written reviews with reviewer's comments, but occasionally only the comments are supplied so as the authors cannot use the numerical scores to argue (for example when the scores given by one reviewer are significantly higher that the scores of other reviewers).

Certain conferences also have an "offline" discussion amongst the reviewers, which is not accessible by the authors, usually moderated by a track chair.

Based on the reaction of the authors of the papers to the initial reviews the reviewers have the opportunity to adjust their reviews (or they should at least make it clear that they have read the author's response).

[1] Information and insights in this section are partly based on public information on the websites of the conferences and partly on private communication with the thesis' supervisor, who has insider experience with the processes of these conferences.

The track chair also can write meta reviews, which serve as summaries of the more in-depth reviews by the other reviewers. That is a standard step in conferences where there is also a conference chair above the track chair who makes the final decision about a paper acceptance. In that case the conference chair mostly decides based on the conclusion given in the meta review, it is quite rare for them to decide otherwise.

In terms of anonymity in the reviewing process, there are many different options and situations. Usually the author is not given the names of reviewers, but reviewers can see the names of the authors. Occasionally conferences follow a double-blind model, where the paper's authorship is anonymized. However, that is not always possible, for example resource tracks (tracks aimed at sharing resources including datasets, software frameworks, ontologies, methodologies or metrics) are just about impossible to anonymize due to the fact that the resources the papers refer to need to be publicly available and in use. Sometimes conferences however do allow the authors to know the names of the reviewers (albeit the reviewers usually have the option to stay anonymous if they wish).

Certain conferences also keep the reviewers anonymous from one another or from the meta-reviewer. The goal here is to give less experienced reviewers a chance to express themselves without worrying about the opinion of senior reviewers and vice versa to keep more experienced reviewers from undermining the opinions given by a less experienced reviewer.

3.3 The structure of conference paper reviews

Different conferences structure their reviews differently. Some are in the form of a completely unstructured text, while some clearly separate the comments in the reviews by the criteria.

For example, the structure of the 2018 EKAW conference reviews is that there is a set of criteria which are assigned numeric scores (relevance, overall evaluation and reviewer's confidence) followed by two yes or no scoring (best paper candidate and poster & demo candidate). It is then followed by a summary of the reviewed paper. The reviewers comments are divided into three parts: *Reasons to accept*, *Reasons to reject* and *Overall evaluation*. The reviews also sometimes contain confidential remarks for the program committee.

The 2017 ISWC and 2018 ISWC conferences were more detailed with its numerical scoring, assigning these scores to:

- Reviewer's confidence
- Appropriateness
- Clarity and quality of writing
- Related work
- Originality/innovativeness
- Impact of ideas and results
- Implementation and soundness

- Evaluation
- Overall paper evaluation

Following these numerical scores, the rest of the ISWC reviews are in the form of unstructured text, unless the author of the review specifically made the decision to structure their comments either by the judged aspect or by the positivity or negativity of their comments.

The most structured reviews available to study were from the 2018 ESWC conference. There, numeric score were assigned to:

- Relevance to ESWC
- Novelty of the proposed solution
- Correctness and completeness of the proposed solution
- Evaluation of the state-of-the-art
- Demonstration and discussion of the properties of the proposed approach
- Reproducibility and generality of the experimental study
- Overall score

while the textual part of the review was clearly divided into the same categories.

3.4 Previous research on conference paper reviews

While there is quite a lot of existing research regarding information extraction from research papers, mostly used for paper summarization and extraction of keywords, not much research was found that focused on information extraction or opinion mining from the reviews of such papers.

In terms of possible generic metrics, previous research on this topic was done as part of a paper on the possibility of pictorial representation of review scores [43]. The metrics they propose are these:

- Relevance
- Novelty
- Technical quality
- State of the art
- Evaluation
- Significance
- Presentation

The overview of different metrics of nine conferences with focus on semantic technology and knowledge engineering and their mapping onto a set of generic criteria can be found in Table 3.1.

Table 3.1: Proposed mapping between generic review metrics and form fields of KE conferences [43]

Review metric	ECAI (2016)	EKAW (2020)	ESWC (2018)	FOIS (2016)	IJCAI (2019)	ISWC, SEMANTiCS (2018)	KR (2014)
Relevance	Relevance	NA	Relevance to ESWC	NA	Relevance	Appropriateness	Relevance of the paper to KR
Novelty	Originality	Novelty	Novelty of the proposed solution	Novelty or innovation	Originality	Originality / innovativeness	Novelty of the contribution
Technical quality	Technical quality	Technical soundness and depth	Correctness and completeness of the proposed solution; Demonstration and discussion of the properties of the proposed approach	Scientific or technical quality	Technical quality	Implementation and soundness	Technical quality
State of the art	Scholarship	NA	Evaluation of the state-of-the-art	References	Scholarship	Related work	Discussion of related work
Evaluation	NA	NA	Reproducibility and generality of the experimental study	NA	NA	Evaluation	NA
Significance	Significance	NA	NA	NA	Significance	Impact of ideas and results	NA
Presentation	Presentation quality	Clarity and quality of writing	NA	Presentation	Clarity and quality of writing	Clarity and quality of writing	Quality of the presentation

One existing research was specifically focused on sentiment analysis of reviews of scientific publications. In this work a dataset of eleven reviews, each of which was manually annotated by their respective authors, was used. The aspects of the reviews they focused on were syntax, style and content. Each reviewer was asked which of these aspects a specific comment in their review focused on, whether the comment was positive or negative, whether an action by the author of the paper was required or just suggested, what was the impact for the overall quality of the paper and whether the author addressed the point raised in the comment. [44] In the eleven reviews there was a total of 421 review comments, most of which (around 44 %) targeted a paragraph or an even smaller part of the paper, almost 30 % were about the paper as a whole and 27 % of comments focused on a section of the paper. The first fairly interesting outcome of this study was that the authors of the study also annotated the review comments themselves and they gathered annotations from peer reviewers. They compared the results and the level of disagreement using a variation of the Mean Squared Error metric, calculated as the square root of the mean squared differences between the average responses of the groups for numerical dimensions and as thee the squared differences from the ratio for each category separately for nominal dimensions. From the results it was clear that *"the model experts and the peers always agree with each other more than they agree with the ground truth in the form of the original reviewer"* [44, p. 6] which according to the authors seems to indicate that *"they misinterpret the review comments in a relatively small but consistent manner"* [44, p. 6].

The result of the annotating phase done by the model experts (the authors) (see Figure 3.1) shows that most of the review comments are about the content of the paper, with much smaller percentages being comments about the style or the syntax. Also the amount of negative comments far exceeds the number of neutral or positive comments.

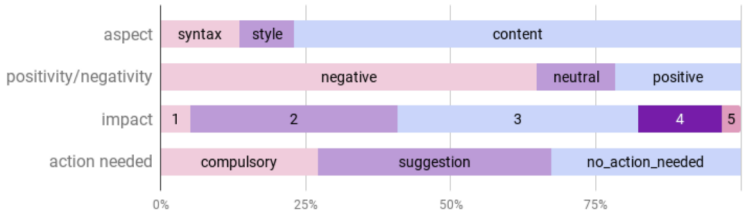

Figure 3.1: The results of the model expert annotations [44]

They have applied 18 different lexicon-based sentiment analysis tools to compare the results and found that the best performing tool was the SOCAL method [45] with a maximum accuracy of 72.8 %. Most of the methods performed quite poorly according to the authors, however they discovered that the methods with more complex rules performed the best, even if the size of their sentiment lexicon was not large. It is important to note in the context of this work that the sentiment analysis was done separately from the aspects purely focusing on the polarity of the comment and they did not develop or used any tools to automatically determine the aspect of the comment.

Another research that is focused on the processes of scientific publishing and more importantly

peer reviewing of these publications builds off of the last paper, however this study is focused on creating a unified model for representation of publications and their assessments *"as well as the involved processes, actors, and provenance in general"* [46, p. 1] in the format of linked data. Their vision is that to give more context to reviews, by linking them to other data, such as information about the reviewer and about the author of a paper, as well as provide a way to link specific parts of a review to the part of the paper they comment on.

In their study they performed a user study based on a set of 7 competency questions to determine the practicality of the ontology they propose. The respondents were asked to judge the importance of each of these questions on a scale from 1 to 5 (1 meaning *not important at all* and 5 *very important*). One of these questions, was *"What is the distribution of the review comments with respect to whether they address the content or the presentation (syntax and style) of the article?"* which was given the average importance score of 3.64, the second highest importance score in their set of questions. This implies that the focus of this paper might indeed be valuable for the community, as it focuses on an even more fine-grained aspect-based sentiment analysis of the reviews.

4. Chosen methods of aspect-based sentiment analysis

The next few chapters are about the preprocessing of data and implementation of all the necessary steps of the aspect-based sentiment analysis. In this brief chapter the goal is to describe the chosen methods that were used to give the reader an overall idea of how the sentiment analysis system works before each step is explained in more detail.

Instead of using traditional machine learning methods, the more linguistic-based methods were used as machine learning methods are, as was previously mentioned, not well suited for the task of sentiment analysis on the aspect level. Also, not much research was done in terms of aspect based analysis over conference reviews, and the more linguistic-based methods allow a more hands-on exploration of the data in this specific domain.

The sentiment analysis method is a variation of the holistic lexicon-based approach (see section 1.2.2), which expects an existing list of aspect expressions as well as a sentiment lexicon. To extract aspect expression, the taxonomy approach is used (as explained in section 1.3.2), where the aspects at the top of the hierarchy represent the criteria and all the terms at the second level represent aspect expressions belonging to the criteria. The chosen set of criteria is as follows:

- Relevance – relevance of the work to the conference
- Novelty – covers novelty of the work as well as its significance and impact
- Technical quality – the technical quality of the work
- State of the art – if proper research was done on the topic
- Presentation – how well written the paper is, grammar
- Evaluation – if the evaluation was carried out properly

As some criteria were hard to distinguish even for human annotators, novelty is designed to also cover similar criteria such as significance and impact. Novelty and significance (or impact) may not be equivalent in meaning, however they are often related and influence one another (for example in sentences such as *"In my opinion though the paper does not have a scientific contribution but is a guide. . . ."*) therefore for simplicity they were joined into one criterion.

In order to create a sentiment lexicon for the domain of conference reviews, the Naïve Bayes classifier (see section 1.2.1) is trained and its output is used to get a lexicon of sentiment/opinion words alongside their polarity.

The holistic lexicon-based approach is also combined with the sentimentr method (described in section 1.2.2), which is more complex in the handling of sentiment modifiers than the holistic approach by itself. Instead of using the sentimentr implementation by the authors of

the method, the algorithm is implemented based on the sentimentr description provided by the thesis' supervisor as a study material which is available in the attachments, as it allows for a better control of the input and output. The only significant deviation from the study guide as well as the original sentimentr implementation is that negation only influences words that come after it in the sentence. Although it is possible for negation to inverse the polarity of a word that precedes it in a sentence (consider the phrase *"I think not."*) it is much more common for the negation to come first and it was found that this change in the algorithm works better for the analyzed texts. Also the more common negation with the auxiliary "do" such as *"I do not think... "* puts more emphasis on the personal opinion of the speaker while *"I think not."* is more of a generalized statement [47], and recognizing the reviewer's opinion is the more relevant task.

5. Analyzed data

This chapter explains how the data analyzed in this thesis were gathered and shows the composition of the resulting dataset. It also describes the preprocessing steps which were taken for each specific purpose – the aspect expressions extraction, the creation of sentiment lexicon and the evaluation of the created system.

5.1 Source of data

As was previously mentioned, the data analyzed in this work are reviews of submissions to conferences focused on semantic technology. Namely from five different conferences – EKAW 2018, ESWC 2018, ESWC 2019, ISWC 2017 and ISWC 2018.

The data for each conference were sourced differently, mainly because with the exception of the ESWC 2019 conference, these reviews are not publicly available. All but the ESWC 2019 data were anonymized by their respective providers.

Data from EKAW 2018 were gathered with the help of the programme committee co-chair Chiara Ghidini and the thesis' supervisor. The reviewers were asked to give consent to the use of their reviews, being able to choose between two levels of consent:

- **Level 1**: I am giving a consent to using my EKAW 2018 review text/score in an anonymized form for the purposes of a sentiment analysis study YES / NO
- **Level 2**: I am giving a consent to using my EKAW 2018 text in a snippet published for illustrative purposes, after the elimination of potentially identity-revealing named entities YES / NO

As a result 247 review were obtained, where only 17 reviewers gave permission on level 1, the rest gave permission on both levels.

The data from the ISWC 2017 and ISWC 2018 were gathered in a similar manner, resulting in 11 and 20 reviews respectively.

A small dataset of the ESWC 2018 reviews was also provided by the thesis' supervisor Vojtěch Svátek, consisting of a total of 6 reviews of two different papers.

The ESWC 2019 data was publicly available through a SPARQL endpoint at `https://metadata.2019.eswc-conferences.org/sparql`. The server hosting the SPARQL endpoint recently went down and apparently will not be made available anytime soon, some data was gathered when the server was still functional.

5.2 Data preprocessing

As the preprocessing of the data differs for different tasks of the entire process and often does not follow the traditional steps, this section explains how the preprocessing was carried out during the individual steps of the research.

5.2.1 Data preprocessing for aspect vocabulary extraction

First each review is tokenized into sentences. Then, each sentence goes through word tokenization using the `word_tokenize` function from NLTK and each token is assigned a POS tag with the `pos_tag` function.

Because the reviews are sometimes not correctly formatted the tokenization might lead to a wrong output. For example, there might not be a space after a punctuation symbol (like a period) and as a result the tokenizer does not separate the punctuation symbol from the next word. For that reason all characters which are not alphanumerical or are not a hyphen are replaced by an empty string in each token.

All tokens are then lemmatized using the `WordNetLemmatizer`. No stop words are removed in the preprocessing. As only nouns, noun phrases and adjectives are extracted, there is no significant overlap with any traditional stop words. Also, some stop words, such as prepositions, are necessary for noun phrase identification.

5.2.2 Data preprocessing for sentiment vocabulary extraction

Given the use of the Naïve Bayes classifier for sentiment vocabulary extraction, a dataset consisting of 1000 review sentences (taken from the ESWC 2019 dataset) was created. Because reviewers often express two different sentiments in a single sentence, such as *"Overall it is a very good paper, but there are some limitations."*, sentences like these had to be split in two – the positive and the negative part. As was described in section 1.2.1, the Naïve Bayes classifier treats a text as a bag of words with an assigned label, there is no syntactic or semantic analysis applied and so a dual polarity in a single example would not be appropriately handled.

Each sentence was labeled with either positive or negative sentiment by the author as well as another annotator (independently as to not influence each other). The results of the two sets of annotations were then compared and it was found that the label did not match for 8 sentences. This was firstly because some sentences described both positive and negative sentiment and were not split correctly which led to each annotator choosing a different sentiment for the entire sentence. It was corrected by only keeping a part of the sentence expressing a single opinion polarity. Secondly, the opposing annotations were in some cases a result of a lack of context for the sentence when each sentence is annotated separately. This was fixed

by looking at the original review and choosing the appropriate polarity based on the context. Finally, some sentences were labeled incorrectly due to a simple mistake of the annotator.

The resulting dataset consists of 743 negative and 257 positive sentences, as negative sentences tend to be present far more in reviews.

When creating the lexicon, first all contractions are expanded, then the review is tokenized into words and assigned a POS tag. Because only words with a high enough frequency are kept, it was also decided to remove stop words, based on a custom stop word list. The NLTK corpus also includes a dictionary of stop words, however it includes words that were expected to have a noticeable influence on the polarity of a sentence such as *should* which rarely points to a positive sentiment in reviews. The list of stop words that were used is:

```
['the', 'be', 'of', 'a', 'to', 'and', 'in', 'it', 'i', 'this',
 'that', 'do', 'for', 'on', 'have']
```

The tokens were again lemmatized, with the exception of adjectives. In this task especially adjectives needed to be kept in the same form as they were originally written in the review. For example, the distinction between *good* and *better* might be important for the polarity of the adjective as *better* is more likely to be used in a negative comment such as *"it would be better if you..."* while *good* mostly keeps its positive polarity.

It is important to note that the `WordNetLemmatizer` that is used for lemmatization is based on the WordNet POS tag set, which is significantly smaller that the Treebank tag set that is used for POS tagging. Because the POS tagger uses the Treebank tag set, all POS tags are transformed into WordNet tags using the function in Listing 5.1. For example, all Treebank tags that refer to verbs begin with "V" and they all get transformed into the single tag that WordNet has for verbs.

Listing 5.1: Transformation between Treebank and WordNet POS tag sets

```
def get_wordnet_pos(pos_tag):

    if pos_tag.startswith('J'):
        return wn.ADJ
    elif pos_tag.startswith('V'):
        return wn.VERB
    elif pos_tag.startswith('N'):
        return wn.NOUN
    elif pos_tag.startswith('R'):
        return wn.ADV
    else:
        return wn.NOUN
```

5.2.3 Data preprocessing for evaluation of results

In order to gain understanding of the accuracy of the sentiment analysis method over conference paper reviews, the output of criterion-sentiment tuples found in each sentence is compared with an annotated dataset of 15 reviews from 3 different conferences (5 from each), namely ESWC 2019, ISWC 2018 and EKAW 2018.

The reviews were labeled by two annotators, with criteria and sentiment polarities found in each review comment (instead of each sentence, as often multiple sentence were part of a single comment or idea and labeling them separately would not be a good approach).

Because the labels of a criterion to which the reviewer points to in a comment very often differed across the two sets of annotations (while both annotators mostly agreed about the sentiment), a discussion between the annotators ensued to attempt a consensus. The results of the process can be seen in Table 5.1. It shows the number of comments that were labeled with some criterion by either annotator, how many times the annotations for a criterion or an aspect did not match a how many times the annotators did not reach a consensus during the discussion phase. Out of 136 annotated comments the annotators did not originally agree in 49 cases when it came to the criteria, which is over one third. In 14 out of those 49 cases the annotators did not reach a consensus even after the discussion. For example, regarding the comment *"Then it will be beneficial to provide a justification of the number of entities... used in the experiment."* one annotator argued it should be labeled with the *evaluation* criterion, as the reviewer questions the small data sample used in the evaluation of the work. The other annotator however insisted on the *presentation* label, based on the fact that the reviewer asks for a clarification on the sample size without outright criticizing it. The number of such disagreements is an interesting outcome, as it shows that the comments are often ambiguous in regards to the specific criteria they comment on, even for human annotators.

Table 5.1: Results of the annotation process for evaluation

conference	**number of comments**	**criterion disagreem.**	**consensus on criterion not reached**	**sentiment disagreem.**	**consensus on sentiment not reached**
ESWC 19	38	18	4	1	1
ISWC 18	33	9	4	1	0
EKAW 18	65	22	6	0	0
Total	136	49	14	1	1

6. Implementation of aspect extraction

In order to create a lexicon of terms that represent the chosen set of criteria, to be used for identifying aspect expressions in the reviews, it was decided to use two main approaches. One is the taxonomy extraction mentioned in 1.3.2 and the second one is extraction of frequent words used by the reviewers for different criteria in a text that is already divided by headers into sections for the respective criterion.

6.1 Manually created taxonomy

As described in section 1.3.2, in order to perform taxonomy extraction, first a user defined taxonomy needs to be created. The original idea is that the taxonomy is a hierarchical representation where the top level of the hierarchy represents the feature of an object and the following levels represent the aspects of that feature. Since in this case, it was not necessary to separate the chosen criteria any further this representation was used to have the main criteria in the top level and only have one level underneath the top level, to specify possible aspect expressions for each criterion. These terms serve as a seed for future expansion of the taxonomy by including extracted crude features with enough similarity to these expressions.

You can see this taxonomy in Listing 6.1.

6.2 Crude features extraction

The next step of taxonomy based extraction is to obtain a set of crude features. The next two subsections describe the extraction algorithms inspired by the frequency-based method of aspect extraction (see section 1.3.1).

6.2.1 Extraction of frequent nouns and noun phrases

The first method of extracting terms that are likely to represent a criterion is to extract frequent nouns and noun phrases (NP). For that the content of the file (representing a single review) is tokenized and each token is assigned a Part-of-Speech (POS) tag. Then, to get the nouns and noun phrases, the extracted tuples (token, pos_tag) are parsed to determine the multi-token sequences which represent nouns and noun phrases.

The RegexpParser from the NLTK library was used for the parsing, which utilizes a user defined grammar, consisting of labeled regular expression rules, describing the sequence of

Listing 6.1: Manually created taxonomy for aspect extraction

```
{
    'relevance': {
        'appropriateness', relevance'
    },
    'novelty': {
        'originality', 'innovativeness', 'innovation',
        'novelty of contribution', 'novelty', 'impact',
        'significance'
    },
    'technical quality': {
        'scientific quality', 'implementation', 'soundness',
        'technical quality'
    },
    'state of the art': {
        'scholarship', 'references', 'related work',
        'state of the art'
    },
    'evaluation': {
        'reproducibility', 'evaluation', 'evaluate', '
           evaluating'
    },
    'presentation': {
        'clarity', 'quality of writing', 'presentation',
        'typo', 'description', 'describe', 'written'
    },
}
```

POS tags we want to assign the label to. The result of the parsing is a tree structure, where each sequence corresponding to the regular expression is labeled accordingly, so this allows us to pick the subtrees labeled by the parser as noun phrases.

The code snippet which shows the defined grammar for NP extraction can be seen in Listing 6.2. The grammar uses POS tags, so NN are nouns, IN are prepositions and JJ are adjectives. The first regular expression, labeled NBAR, searches for nouns and adjectives, terminated with nouns, allowing us to discover phrases like *black box*, where the meaning changes when we consider both of these words separately. The second regular expression, labeled NP, looks for NBAR expressions connected with prepositions such as *of*, *in* etc. [48]

Listing 6.2: Grammar for the extraction of noun phrases.

```
RegexpParser( """
        NBAR:
            {<NN.*|JJ>*<NN.*>}

        NP:
            {<NBAR>}
            {<NBAR><IN><NBAR>}
 """)
```

In order to then extract only the wanted noun phrase sequences, the tree is traversed. For each subtree, labeled as NP, each token of the sequence is lemmatized, it is checked for if its length is at least two characters, but less than 20 characters, and if so, the lemmatized tokens are joined into a single string and appended to the list of NPs in the file.

By performing this extraction with each file, the result is a list of lists, where each list represents the extracted nouns and noun phrases from one file. To then obtain the ones that are frequent enough across all the reviews, and may therefore represent the criteria, the support of a noun phrase across the reviews is calculated. The following equation represents the calculation of the support metric for a word w_i where N_{w_i} is the number of reviews containing the word w_i and N is the total number of reviews:

$$support(w_i) = \frac{N_{w_i}}{N} \tag{6.1}$$

In order to perform this calculation the data is transformed into a matrix, where each row represents one review, each column represents a criterion expression candidate. The values of an element in *row-i* and *column-j* is 1 if the criterion expression candidate j is present in review i or 0 if it is not. To get the support of a criterion expression candidate j, the sum of *column-j* is divided by the total number of rows. If the support is greater than the minimum support, the criterion expression candidate is kept, if not, it is discarded. Through various experiments it was found that the best value for minimum support is 2 %, which gives us reasonable candidates for criterion expressions, but does not include too many candidates to

make the process of manually confirming them too tedious. It may seem like a very "minimal" minimal support, however the number of reviews that were at disposal to extract frequent noun phrases was fairly small, so a small support was necessary to account for that fact. It should not be too big of an issue though, considering that the candidates are further tested to determine their similarity to the manually created taxonomy, which reduces the number of criterion expression candidates the user has to go through.

6.2.2 Extraction of frequent adjectives

Although the original frequency-based algorithm (see section 1.3.1) only focuses on nouns and noun phrases, by going through the training data, it was discovered that fairly often, the criterion expression found in the reviews take the form of adjectives.

Consider the phrase *"The topic addressed by the paper relevant to the conference."*. Here, the adjective *relevant* evidently corresponds to the criterion *relevance*. Because of that, it was decided to extract frequent adjectives from the reviews as well and again calculate the support of each adjective across the reviews with the same technique as was used with the noun phrase extraction.

6.3 Extraction by review structure

The data from the 2018 ESWC conference that was obtained had the review text divided into sections where the different sections represented the different criteria. The structure of these reviews can be seen in Figure 6.1, where in bold are the structural part of the review format which stays the same across all reviews.

It was decided to leverage this data to extract frequent words from each one these sections across the reviews. The frequent words of each section are included in the new aspect expression taxonomy directly, they do not go through the same process of similarity matching against the manually created taxonomy as the candidates that were chosen purely on their frequency. This is useful because it allows to extract new possible criterion terms that would not match with any of the terms in the taxonomy, which were originally not thought to be included.

Each term frequent enough in a criterion section across all the reviews is included in the taxonomy based on a match between the ESWC set of criteria and the chosen set of metrics. The mapping between the two sets of metrics was done according to Table 3.1.

The aspect expression candidates created by this method are still evaluated by the user as explained in section 6.5.

6.4 Similarity matching against the manually created taxonomy

After a set of crude features is obtained, the next step is to map these features to the user defined taxonomy. As previously described, the similarity of a crude feature to the aspect expression in the taxonomy needs to be calculated. If the feature is similar enough, it is passed to the final step of the taxonomy extraction, which is an interactive revision process. This process is described in more detail in the next section.

For measuring the similarity between two terms, it was determined to use the WordNet tool and its `path_similarity` metric which returns a score denoting how similar two word senses are, based on the shortest path that connects the senses in the hypernym hierarchy. The score is in the range 0 to 1 where 1 denotes identity (when word is compared to itself). [33]

The main obstacle of using this metric is that it does not work well with adjectives. That is because all nouns are part of one big hierarchy, but that is not the case for other parts of speech such as an adjective, an adverb etc. So for example the similarity for the words "relevance" and "relevant" is zero, even though the words are closely related. Because the extracted crude features may contain adjectives (in the case of noun phrases) or be adjectives themselves (in the case of frequent adjectives extraction), a workaround was implemented by transforming the adjectives to their closest related noun, and perform the similarity measurement on these nouns. If the similarity of these nouns is greater than the similarity threshold, the original (albeit lemmatized) word is passed on.

Another issue is measuring similarity with terms consisting of multiple words. Certain multi-token terms are already present in the WordNet thesaurus (such as *state of the art*), and getting their synsets to perform similarity matching is as simple as replacing the spaces between words with underscores. However, some multi-token words included both in the user defined taxonomy and in the crude features cannot be found in WordNet directly. It was solved by calculating the maximum similarity between each token of one term to all the tokens of the other term. Of these similarities the maximal one is chosen.

The final pseudocode for similarity measuring between two terms is in Figure 6.2. When comparing frequent adjectives to the taxonomy, the part-of-speech argument is set accordingly.

Another issue of wordnet's `path_similarity` is that it is asymmetrical. So sometimes `path_similarity(x,y)` returns `None` or 0 while `path_similarity(y,x)` returns a non-zero value. This is because for some words, a fake root in the hierarchy may be added to find a path between to two words, but this depends on the order in which the two words are supplied to the `path_similarity` function. Therefore the final calculation of similarity between two terms term1 and term2 is defined as the maximum between similarity(term1,term2) and similarity(term2,term1).

The threshold for similarity of a term to the taxonomy was set to 0.3. Therefore every term with a similarity to any term in the manually created taxonomy equal or greater than the

threshold will become a criterion expression candidate under the same criterion as the term it was most similar to.

6.5 User validation of final taxonomy

When aspect expression candidates are generated and sorted by the criterion they most likely represent they have to pass the final validation. This validation is a manual process, where a user goes through every criterion expression candidate and decides between three options:

- The criterion expression candidate is added under the criterion which was algorithmically determined as the most probable.
- The user disagrees with the most probable criterion and sorts the criterion expression candidate under a different criterion.
- The user decides not to include the criterion expression candidate in the taxonomy at all.

The interactive process of the user validation of the final taxonomy can be seen in Figure 6.3.

6.6 Resulting taxonomy of aspects

The entire process of aspect expressions extraction and user validation (carried out by the thesis' author) resulted in 57 new terms in the taxonomy. The new terms added under each criterion are showcased in Table 6.1 along the original terms from the manually created taxonomy.

Table 6.1: Result of the aspect expression extraction

criterion	**aspect expressions - old**	**aspect expressions - new**
relevance	appropriateness, relevance	important topic, relevant, contribution, topic
novelty	originality, innovativeness, innovation, novelty, novelty of contribution, impact, significance	originality innovativeness, scientific contribution, improvement, novel, idea
technical quality	scientific quality, implementation, soundness, technical quality	running example, scalability, code, design, usability, implementation and soundness, technical detail
state of the art	scholarship, references, related work, state of the ar	reference, related work section, references, benchmark, comparison, previous work, related research
evaluation	reproducibility, evaluation, evaluating, evaluate	evaluation section, coverage, experimentation, score, experimental result, experimental, experimental evaluation, support, empirical evaluation, accuracy, assessment, evaluation result, recall, metric, experiment
presentation	clarity, quality of writing, presentation, typo, written, describe, description	english, scientific paper, notation, text, last sentence, sec, write, figure, introduction, document, explained, current form, reading, writing, first paragraph, intro, readability, format, paragraph

Relevance to ESWC:
numerical score (score interpretation)
Novelty of the Proposed Solution:
numerical score (score interpretation)
Correctness and Completeness of the Proposed Solution:
numerical score (score interpretation)
Evaluation of the State-of-the-Art:
numerical score (score interpretation)
Demonstration and Discussion of the Properties of the Proposed Approach:
numerical score (score interpretation)
Reproducibility and Generality of the Experimental Study:
numerical score (score interpretation)
Overall score:
numerical score (score interpretation)
Reviewer's confidence:
numerical score (score interpretation)
Open Reviewing Opting Out:
numerical score (score interpretation)
Overall evaluation (*Resources and In-Use tracks only*, Research reviewers please only put "n/a"):
numerical score (score interpretation)
———— **Relevance to ESWC** ————
reviewer's comment
———— **Novelty of the Proposed Solution** ————
reviewer's comment
———— **Correctness and Completeness of the Proposed Solution** ————
–
reviewer's comment
———— **Evaluation of the State-of-the-Art** ————
reviewer's comment
———— **Demonstration and Discussion of the Properties of the Proposed Approach** ————
reviewer's comment
———— **Reproducibility and Generality of the Experimental Study** ————
reviewer's comment
———— **Overall score** ————
reviewer's comment

Figure 6.1: ESWC 2018 review structure

```
Data: term 1, term 2, part-of-speech
Result: the similarity of the term as a numeric score between 0 and 1
for both terms do
    if the term is just one word then
        term_synsets = find all synsets of the term ;
        if the part-of-speech is adjectives then
            term_synsets += find all synsets of the closest related noun
    else
        term_underscored = substitute all spaces in the term with underscores ;
        term_synsets = find all synsets of term_underscored ;
    if no synsets are found for a multi-word term then
        term_synsets = synsets of all words in the term (including transformed
         adjectives) ;
score = maximum similarity between all term_synsets of term1 and term_synsets of
 term2 ;
return score
```

Figure 6.2: Algorithm for similarity measurement between two terms

```
Does the term "topic" belong under aspect "relevance" ? [y/n]
y
Does the term "relation" belong under aspect "relevance" ? [y/n]
n
Does it belong under any of these aspects ?:
relevance                [a]
novelty                  [b]
technical quality        [c]
state of the art         [d]
evaluation               [e]
presentation             [f]

none of the above        [n]
n
Does the term "introduction" belong under aspect "novelty" ? [y/n
   ]
n
Does it belong under any of these aspects ?:
relevance                [a]
novelty                  [b]
technical quality        [c]
state of the art         [d]
evaluation               [e]
presentation             [f]

none of the above        [n]
p
```

Figure 6.3: The interactive process of user validation of proposed aspect taxonomy.

7. Creation of sentiment lexicon

Through some initial experimentation with various existing sentiment lexicons such as the SenticNet sentiment lexicon and the NLTK's SentiWordNet sentiment lexicon it was discovered that these universal sentiment lexicons are not well suited for application on the domain of conference paper reviews.

One issue is that both of these sentiment lexicons assign sentiment polarity on a scale. Therefore most dictionary words have a certain sentiment polarity which was considered inappropriate in this task, as words which would generally be considered neutral should not somehow skew the polarity of words that might actually be important. These words often have a polarity around the center of the polarity interval (for example if the polarity is assigned on a scale from -1 to 1 they usually have polarity somewhere around 0) and could be removed by using some polarity threshold. However, that might also lead to losing some words that are actually important for sentiment classification in this domain. For example in SenticNet, the word *clarification* has polarity of -0.09, but it is often used in sentences such as *"The section about experimental results needs some clarification"* where the sentiment is clearly negative.

Another possible issue with pre-made sentiment lexicons is that they mostly do not include punctuation. However, punctuation might have great semantic significance in conference paper reviews, as they are often written in plaintext and punctuation is used to compensate for the lack of usual formatting styles such as bullet points.

For that reason, it was decided to create a custom, domain-specific sentiment lexicon. This chapter describes the process of compiling a sentiment lexicon from a set of annotated sentences taken from reviews using the Naïve Bayes classifier.

7.1 Implementation of sentiment lexicon generation using Naive Bayes

The Naïve Bayes classifier, as described in section 1.2.1 is a probabilistic classifier which needs a training dataset of labeled data in order to determine the influence of different evidence on the class. The obtainment of this dataset is described in section 5.2.2.

NLTK's implementation of the classifier was used. Of all the lemmatized tokens in the reviews, the number of tokens the classifier needs to process was limited to 450. The tokens are considered features by the algorithm, each token being transformed into a column where the value of the column for each row representing a review is true or false depending on the presence of the token in the review. This was achieved thanks to the `FreqDist` class from the NLTK's `probability` module.

The dataset of labeled review sentences was split to have a testing dataset of 50 example

sentences to determine the accuracy of the classifier and the rest of sentences was used for training. The test dataset is fairly small, but it was considered far more preferable to leave most examples to the training dataset to produce a hopefully more accurate sentiment lexicon, even though the trade-off is not being able to judge the created lexicon in great detail in this phase. That being said, the accuracy of the classifier on the testing dataset was 0.78, meaning 78 % of sentences were classified correctly.

The classifier allows us to get a list of features which have the highest contribution to classification through its `show_most_informative_features` method which based on the number we specify as its argument outputs a list of features with their ratio of occurrences in negative and positive sentences. The output when applied to the training data can be seen in Table 7.1. It shows that it is more than 35 times more likely that the word *easy* occurs in a sentence labeled as *positive* while a question mark occurs far more often in negative sentences.

Table 7.1: Output of the Naïve Bayes classifier

Most Informative Features		
easy = True	positi : negati =	35.8 : 1.0
interesting = True	positi : negati =	15.3 : 1.0
but = True	negati : positi =	14.8 : 1.0
topic = True	positi : negati =	13.7 : 1.0
? = True	negati : positi =	12.3 : 1.0
what = True	negati : positi =	10.9 : 1.0
sound = True	positi : negati =	10.6 : 1.0
community = True	positi : negati =	9.9 : 1.0
not = True	negati : positi =	8.9 : 1.0
idea = True	positi : negati =	8.1 : 1.0
interest = True	positi : negati =	7.9 : 1.0
good = True	positi : negati =	7.5 : 1.0
clearly = True	positi : negati =	7.4 : 1.0
well = True	positi : negati =	7.4 : 1.0
me = True	negati : positi =	7.2 : 1.0
bring = True	positi : negati =	6.8 : 1.0
valuable = True	positi : negati =	6.4 : 1.0
why = True	negati : positi =	5.6 : 1.0
conference = True	positi : negati =	5.6 : 1.0
write = True	positi : negati =	5.5 : 1.0
effort = True	positi : negati =	5.4 : 1.0
highly = True	positi : negati =	5.2 : 1.0

The `show_most_informative_features` method was then adjusted to create a function which transforms these ratios of occurrences in sentences with positive or negative sentiments into a sentiment lexicon.

Each of the most informative tokens is given a value of -1 or +1 depending on if they occur

more often in negative or positive sentences (where -1 corresponds to negative sentiment and +1 corresponds to positive sentiment).

7.2 Created sentiment lexicon

From the list of most informative features, the top 100 words were chosen to be included to the sentiment lexicon (setting the ratio threshold at 2.4 : 1.0). Then the results were compared with the SenticNet sentiment lexicon, to see what is the level of agreement between the two lexicons and it was discovered that in 13 cases, the polarity of the sentiment of words found in both lexicons differed and in 31 cases a word from my lexicon was not found in SenticNet. Surprisingly not all the words that were not found in SenticNet were not found due to the aforementioned lack of punctuation in SenticNet or because these words could truly be considered neutral. SenticNet was missing some words which are considered fairly meaningful in sentiment analysis such as *rather* or *should*.

A list of 40 positive and 66 negative words compiled manually during the process of labeling the training dataset was also included. The resulting sentiment lexicon contains 186 sentiment words out of which 88 have positive polarity of +1 and 98 have a negative polarity of -1.

8. Implementation of aspect-based sentiment analysis for conference paper reviews

This chapter explains how the final algorithm for sentiment analysis was implemented, using the results of criterion expressions extraction and creation of the sentiment lexicon to build a lexicon-based method for aspect-based sentiment analysis.

8.1 Design of classes

To get a more detailed insight in the back-end of the sentiment analysis algorithm, this section describes how the data is processed and stored in classes to keep track of all necessary attributes that might be associated with each object such as the review itself or a particular sentence.

8.1.1 Review

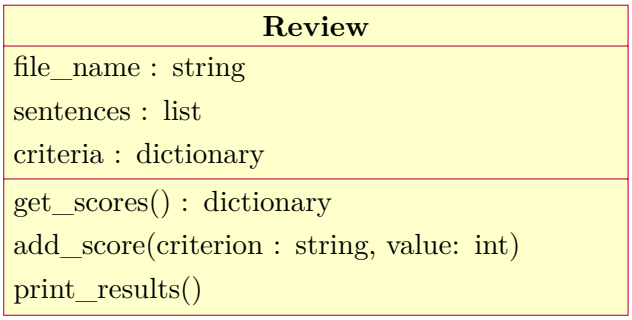

Figure 8.1: Review class diagram

Each review is represented by the `Review` class. When the class is initiated with its constructor the attribute `file_name` is initialized with the file_name of the review (for the purpose of printing the results). The review text is the tokenized into sentence using NLTK's `sent_tokenize` function. To keep track of the numerical scores of the review, which are assigned through the process of sentiment analysis, there is a dictionary `criteria`, in which the keys of the dictionary are the set of generic criteria as chosen in chapter 4. The values are represented by the `CriterionScore` class.

The class also has three methods. The `add_score` method accepts the name of a criterion and a sentiment value (-1 or +1) that should be added to the criterion and updates the `criteria`

dictionary accordingly. The `get_scores` function returns the final scores for a review in the form of a dictionary where the different criteria are the keys and the values are the numerical scores normalized between 1 and 5. Finally the `print_results` function outputs the scores of the review onto the command line in a format shown in Figure 8.2.

```
row233.txt
        relevance                       3
        novelty                         5
        technical quality               1
        state of the art                5
        evaluation                      2
        presentation                    1
```

Figure 8.2: Example output of the `print_results` method of the `Review` class

8.1.2 Sentence

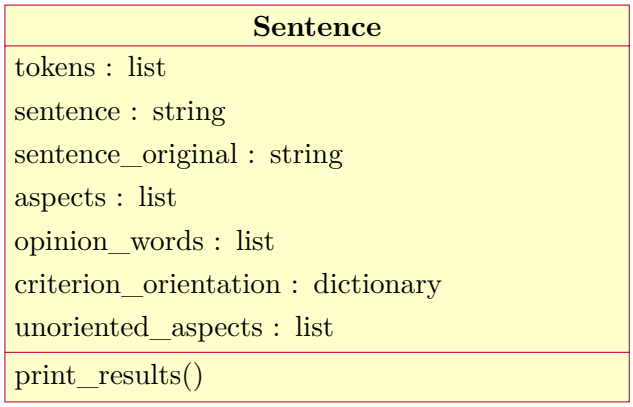

Figure 8.3: Sentence class diagram

The `Sentence` class represents one sentence from a review. When initialized the original sentence is kept in the `sentence_original` attribute for printing purposes, however for the needs of the sentiment analysis the sentence is also tokenized and lemmatized after all contractions are expanded. The `MWEtokenizer` was decided to be used here as well as NLTK's `word_tokenize` function. The `MWEtokenizer` takes a string in the form of a list of tokens and retokenizes it, *"merging multi-word expressions into single tokens, using a lexicon of MWEs"* [49]. The reason is that because some criterion expression are multi-words expressions, such as *related work* they need to be kept as single tokens, in order for them to be recognized when matching the sentence against the aspect taxonomy. Therefore each criterion expression in the taxonomy is added as a multi-word expression to the `MWEtokenizer` lexicon as well as some of the contraction expansions.

The tokens are also lemmatized, with the exception of adjectives to keep words such as *better*

to be lemmatized to *good* (the reasoning behind that was explained in section 5.2.2). The tokenized and lemmatized sentence is then kept in the `tokens` attribute of the class and the lemmatized tokens are stringed back together in the `sentence` attribute.

For the sentiment analysis algorithm it is also essential to know which aspect expressions and which sentiment words the sentence contains. To get a list of aspect expressions the `Taxonomy`'s class method `get_aspects` is called, which compares the tokens it gets as an argument with the taxonomy of criteria expressions and returns the matches as a list of the `Aspect` class objects. The list of aspect expression is kept in the `aspects` attribute of the class.

To get a list of sentiment words the `find_opinion_words_sentiment` function is called, which belongs to the sentimentr module. This function gets a list of tokens as an argument and then for each token that is found in the sentiment lexicon it calculates the polarity value using the sentimentr method (described in section 1.2.2). It returns a list of (sentiment word, polarity) tuples, which are then used to initialize the `OpinionWord` class. However, if there is an overlap between a sentiment word and a criterion expression, the word is at this point discarded (it may be used in the future, if the orientation of an aspect expression is not found). The list of sentiment expressions is kept in the `opinion_words` attribute.

If the main part of the sentiment analysis algorithm fails for an aspect expression (it evaluates the polarity at zero) there is a set of rules which try to determine the sentiment in some other ways. The list of aspect expressions which need the further evaluation is kept in the `unoriented_aspects` attribute to which these aspects are continuously added as the review is analyzed.

To keep a track of which criteria a sentence is focused on as well as the polarity of the opinion that is expressed, there is the `criterion_orientation` attribute, which is a dictionary where the keys are the criteria and the values are numerical scores.

The `Sentence` class contains a single method, `print_results`, which allows the user to see the results of the analysis on a more fine-grained level than the `print_results` method of the `Review` class. It prints the original sentence and for each criterion in the sentence shows if the polarity of the opinion on the criterion is positive, negative or neutral.

8.1.3 OpinionWord

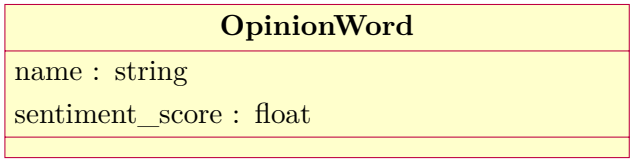

Figure 8.4: OpinionWord class diagram

The `OpinionWord` class represent an opinion/sentiment word with the name of the word in the `name` string attribute and the sentiment polarity determined by the sentimentr algorithm kept in the `sentiment_score` attribute as a decimal value.

8.1.4 CriterionScore

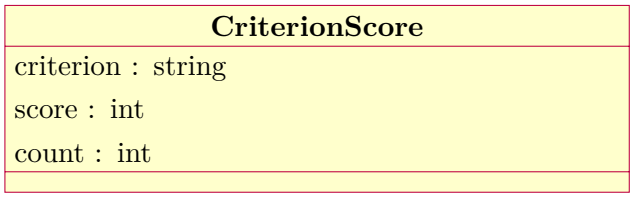

Figure 8.5: CriterionScore class diagram

The `CriterionScore` class is to keep track of the six main criteria which the reviews are evaluated by. The `criterion` string attribute is assigned the name of the criterion, the `score` attribute keeps track of the numerical score of the criterion and the `count` attribute counts the number of times new value is added to the `score`. Any time an aspect expression belonging to the criterion is discovered in a sentence and its orientation is evaluated by the sentiment analysis algorithm it is added to the `score` attribute so the `count` attribute allows as to count the average value so that all scores are normalized.

8.1.5 Taxonomy

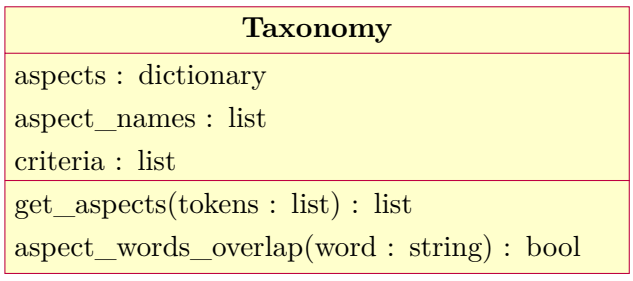

Figure 8.6: Taxonomy class diagram

The `Taxonomy` class represents the taxonomy created in chapter 6. The class is initiated only once, when the taxonomy is loaded from the JSON file in which it was saved during the taxonomy generation phase. The `aspects` attribute is a python dictionary which follows the same structure as the JSON (see Figure 6.1). For simplicity of other operations, such as finding aspect expressions in a sentence, the `criteria` attributes keeps a list of all the main criteria/metrics, while the `aspect_names` attribute keeps track of all aspect expressions in the taxonomy.

There are two method in the class, the method `get_aspects` accepts a list of tokens in a sentence, finds if any of the tokens are aspect expressions and if so, returns a list of the `Aspect` class objects that correspond to them. The second method `aspect_words_overlap` accepts a string as an argument and returns *True* if the string overlaps with a string of an aspect expression and *False* otherwise.

8.1.6 Aspect

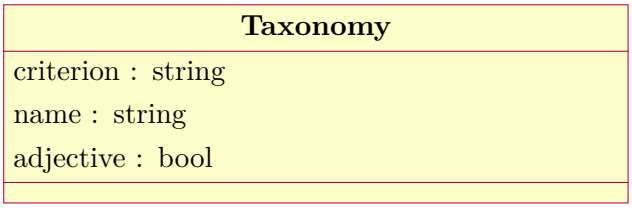

Figure 8.7: Aspect class diagram

The `Aspect` class simply represents an aspect expression, the name of which is saved in the `name` attribute. It also keeps the name of the criterion/metric under which the aspect expression belongs in the `criterion` attribute, in the form of a string.

If the aspect expression is an adjective the `adjective` attribute is set to *True* to enable some special handling of these expressions. The value of `adjective` is determined by finding the WordNet synsets of the expression and if the POS tag of any of the synsets is that of an adjective. In NLTK's wordnet implementation that means the tag is either *a* for adjectives or *s* for satellite adjectives which are adjectives without a direct antonym.

8.2 Description of the algorithm

The aspect-based sentiment algorithm that was created for the task of extracting criteria orientations from a text is loosely based on the algorithm mentioned in section 1.2.2, but some changes were made. The main part of the algorithm is shown in Figure 8.8.

```
Function opinion_orientation(review):
    for each sentence s_i in review that contains a set of aspect expressions do
        if the number of aspect expressions in s_i > 5 then
            continue;
        for each aspect a_j in s_i do
            orientation = 0;
            for each opinion word ow_k in s_i do
                if adversative_between_ow_and_aspect(ow_k, a_j, s_i) then
                    continue;
                distance = distance_betweeen_words(ow_k, a_j, s_i);
                orientation += ow_k.sentiment_score / distance;
            orientation = orientation_to_interval(orientation);
            if orientation != 0 then
                a_j's criterion orientation in s_i += orientation;
                review.add_score(a_j's criterion, orientation);
            else
                add a_j to s_i's unoriented_aspects;
        for aspect a_l in s_i's unoriented aspects do
            orientation = 0;
            opinion_words = find_opinion_words_sentiment(a_l);
            for ow_i,pol in opinion_words do
                if ow_i==a_l.name and a_l is an adjective then
                    orientation = pol;
                    break;
            if orientation==0 then
                orientation = apply_intra_sentence_rules(s_i, a_l);
            orientation = orientation_to_interval(orientation);
            a_l's criterion orientation in s_i += orientation;
            if orientation!=0 then
                review.add_score(a_l's criterion, orientation)
```

Figure 8.8: opinion_orientation algorithm

Function `apply_intra_sentence_rules(s, aspect):`

ow= opinion word closest to *aspect* in *sentence*;
words_between= a list of words between *ow* and *aspect*;
distance=**distance_betweeen_words**(*ow*, *aspect*, *sentence*);
if *adversative in words_between* **then**
  return $-1 \times \frac{ow.sentiment\_score}{distance}$;
else
  return $\frac{ow.sentiment\_score}{distance}$;

Figure 8.9: apply_intra_sentence_rules algorithm

8.2.1 Determining opinion orientation using aspect expressions and sentiment words in a sentence

For each review its sentences (which are here objects of class `Sentence`) are iterated through. If the number of aspect expressions in the sentence is more than 5 the sentence is not evaluated. That is because if the number of aspects is that high in a single sentence it would be hard to evaluate which opinion words belong to which aspect. It is especially an issue with the numerical evaluations at the beginning of reviews, which are sometimes not structured by newlines or any other separator.

Given a sentence s_i that contains a set of aspect expressions, a sentiment polarity score is calculated for each expression. Given a set of sentiment words in the sentence, their collective influence is calculated based on the sentiment score given by the sentimentr algorithm which is divided by the distance of the sentiment word from the aspect expression. These scores are then aggregated by a sum function for each aspect expression.

An expression is assigned the sentiment polarity using the `orientation_to_interval` function. Its default functionality is to return -1 or +1 if the orientation sent as an argument is positive or negative, 0 otherwise. If the optional second argument is set to `False`, it can also return -0.5 or +0.5 if the orientation is in the $]-0.5, 0[$ or $]0, +0.5[$ intervals respectively.

If the polarity is 0, it is added to the list of unoriented aspects of a sentence for further processing, but if the algorithm succeeded in determining the polarity, the score for a criterion under which the aspect expression belongs in the taxonomy is adjusted in the review as well as the sentence.

8.2.2 Adjectives as aspect expressions

When for some aspect expressions the orientation is unknown after aggregating polarities of nearby opinion words, they are added to s_i's list of unoriented aspects. These aspect expressions are then first evaluated using the adjective rule, where if the aspect expression is

an adjective, it may be used as an opinion word itself. That is the case in sentences such as *"This paper is highly relevant to the conference"*, where the adjective *relevant* points to the relevance criterion, but also expresses a positive polarity on said criterion. In cases when the aspect expression is an adjective, its polarity is determined using the sentimentr algorithm as if it was an opinion word, to cover cases such as negation.

8.2.3 Intra sentence rules

If the aspect expression orientation is still unknown after the use of the adjective rule it is evaluated using the intra sentence rules (see Figure 8.9), which rely on the fact that a sentence only expresses one polarity unless it includes an adversative conjunction. For each so far unoriented aspect expression the closest opinion word is found as well as all the words between the aspect and the opinion word. If there is an adversative conjunction in between the opinion word and the aspect expression, that likely means the sentiment polarity was inverted by the conjunction and therefore the aspect expression should be given the opposite polarity of the opinion word. This should help in sentences such as *"The evaluation shows great results but the dataset was small."* in which *dataset* might be an aspect expression pointing to the evaluation criterion but *small* might not be in the sentiment lexicon. For a human, it is easy to point to small as a negative word here based on the context, but that is not always the case, for example in a phrase *small error* it would be positive. The closest identified opinion word in this sentence would be *great*, with a positive polarity, but given the fact that there is *but* in between *great* and *dataset*, the polarity assigned would be negated. If no adversative conjunction is found, the aspect expression is given the polarity of the closest opinion word without any changes, following the *one sentence – one polarity* idea.

8.2.4 Sentences with neutral sentiment

If the orientation of an aspect expression is still 0 after the application of all rules, it is finally evaluated as neutral. Sometimes, this might mean that the sentiment was present but expressed in a way that the algorithm did not recognize. It might also mean a false positive for an aspect identified within a sentence. That is because there might be an overlap between aspect expressions and expression that are often used within the field for describing an idea presented in a paper. Take the sentence *"In this paper authors investigate how state-of-the-art language technologies can be ported to the historical ecology domain."* in which the algorithm would think that the *state-of-the-art* expression points to the state of the art criterion but here it is simply a statement about the objective of the paper.

These aspect expression do not influence the numerical scores of a review, the aspects with neutral orientation are however still shown in the algorithm's output at a sentence level.

9. Evaluation of results

The goal of this chapter is to perform a rigorous evaluation of the created aspect-based sentiment analysis algorithm. First, the numerical scores of each criterion estimated by the algorithm are compared to the scores given to these criteria by the reviewers. Then, the algorithm's output is evaluated in more detail, by establishing the precision and recall of aspect identification and the accuracy of the sentiment analysis, using an annotated set of reviews. Finally, the results of the evaluation are discussed and suggestions are made for future improvements.

9.1 Evaluation using reviews with numerical scores

The reviews from ISWC 2018 contain numerical scores for a wide range of criteria as was mentioned in section 3.3. It was decided to compare the numerical scores outputted by the sentiment analysis algorithm with the ground-truth scores taken from the reviews. Because the ISWC set of criteria is more detailed that the set of criteria the algorithm works with, it was necessary to create a mapping between them which you can see in Table 9.1.

Table 9.1: Mapping between the chosen set of criteria and ISWC 2018 criteria

Algorithm's criteria	**ISWC 2018 criteria**
relevance	appropriateness
novelty	originality/innovativeness
	impact of ideas and results
technical quality	implementation and soundness
state of the art	related work
evaluation	evaluation
presentation	clarity and quality of writing

The numerical output was evaluated using the mean absolute error function (MAE), which measures the absolute average distance between the real data Y and the predicted data $\bar{Y}$:

$$MAE = \frac{1}{n} \times \sum_{i=1}^{n} |Y_i - \bar{Y}_i| \tag{9.1}$$

The MAE was calculated separately for each criterion to see if the algorithm performs better or worse for some of them. The default range of [1;5] for scores outputted by the algorithm was matched to the [-2;2] range of the ISWC reviews. Because the algorithm outputs "n/a" instead of a number for criteria for which no sentiment value was found, the number of times the "n/a" value occurs for each criterion was also calculated.

The results of the numerical evaluation carried over the 20 ISWC 2018 reviews can be seen in Table 9.2. It is clear that the algorithm often struggles with finding any criterion score, especially when it comes to relevance, novelty and technical quality. Even when it does give a numerical score, it is often fairly off. This could have several explanations. Firstly it is possible that when reviewers have the option of expressing their opinion numerically, they sometimes do not feel the need to also give a more elaborate explanation. The second explanation is that the algorithm simply does not perform well when it comes to discovering aspect expressions and/or sentiment words. This is studied more closely in the next section, where the results are evaluated on the sentence level.

Another issue might be the way in which the numerical scores are estimated – each time an aspect expression is discovered and assigned a polarity of +1 or -1 the value is added to the respective criterion score. Finally the scores are averaged by the number of times a value was added and normalized to a given score range. Therefore, if for a criterion only one aspect expression is found with a given polarity the final score will always be an extreme in the score range, but that is a much rarer occurrence in the scores given by human reviewers. To check if this might be the issue it was tested by changing the normalization of polarity to four different values, where a criterion is added a score of +1 if the orientation of an aspect expression is higher than 0.5, a score of +0.5 if the orientation is higher that 0 and analogously the -0.5 and -1 scores for negative orientations. The results after that change can be seen in Table 9.3. It is apparent that this leads to better results and so a more fine-grained approach to polarity is necessary.

Table 9.2: Results of the numerical evaluation of ISWC 2018 reviews

Criterion	**MAE**	**Number of missing values**
relevance	1.44	11
novelty	1.68	0
technical quality	1.64	9
state of the art	1.38	4
evaluation	1.35	3
presentation	0.94	4

Table 9.3: Results of the numerical evaluation of ISWC 2018 reviews using a more granular approach to polarity

Criterion	**MAE**	**Number of missing values**
relevance	1.33	11
novelty	0.93	0
technical quality	1.09	9
state of the art	1.06	4
evaluation	0.82	3
presentation	0.69	4

9.2 Evaluation using annotated review comments

As was stated in section 5.2.3, a dataset of reviews with comments annotated by criterion and sentiment was prepared in order to evaluate the accuracy of the algorithm in more detail. The goal was to determine the precision and recall of the algorithm as well as to take a closer look at where the algorithm struggles with accurate results. This section presents the outcome of the evaluation first for the criterion identification and then for the sentiment analysis.

9.2.1 Evaluation of the criterion identification

The annotated dataset was compared to the output of the sentiment analysis algorithm. Each annotated comment for which the appropriate criterion was found by the algorithm was classified as *true positive* (TP), each comment which was labeled by a criterion but the algorithm did not discover it was labeled by *false negative* (FN) and each time the algorithm outputted a criterion incorrectly (it was not labeled with the same criterion by the annotators) it was classified as *false positive* (FP). Because sometimes a comment was labeled differently by each annotator and they did not reach an agreement during the annotation phase, a comment was labeled as TP if the algorithm outputted either one of the annotated criteria. Also it is important to note that while the annotators labeled comments which sometimes consist of multiple sentences, the algorithm works on a sentence level. Therefore, the outputted sentences and their criteria were grouped for the evaluation to match the comments.

Some of the reviews contain numerical scores for the extracted criteria, however while the algorithm always correctly identifies the respective criteria, it is unable to assign a sentiment polarity based on the score value. Because each conference might use a different scale for these scores and the algorithm should work independently of any knowledge about the specific source of reviews, it was decided against using the scores for the sentiment analysis and instead only focus on the textual data. Therefore the parts of reviews with numerical scores were ignored for both the annotation and evaluation phase.

The results of the comparison between annotated criteria and outputted criteria can be seen in Table 9.4.

Table 9.4: Evaluation of the criterion identification

conference	**TP**	**FP**	**FN**
ESWC 2019	17	15	14
EKAW 2018	31	17	34
ISWC 2018	22	20	13
Total	70	52	61

To quantify the accuracy of the results, first the precision for criteria over the entire dataset

was calculated using the following equation:

$$precision = \frac{TP}{TP + FP} \tag{9.2}$$

which resulted in a precision of 57.38 %.

The recall for criteria over the entire dataset was calculated as:

$$recall = \frac{TP}{TP + FN} \tag{9.3}$$

which evaluates the recall at 53.44 %.

Figure 9.1 shows the contribution of each criterion to each of the evaluation label (TP, FP or FN), while Figure 9.2 shows these results for each criterion individually by displaying the relative amount of each of the evaluation labels to their sum. The algorithm visibly struggles the most with discovering the *technical quality* criterion, where the false negatives amount for 61 % of the assigned evaluation labels, 25 % were false positive and the algorithm succeeded in correctly finding the correct criterion in just 14 % of cases. The algorithm also did not fare particularly well for the *state of the art* criterion, where although the relative amount of false negatives is fairly small, the number of false positives is at 62 %, meaning that a considerable amount of sentences get labeled with this criterion wrongly.

In terms of discovering the correct criteria, the *presentation* criterion amounts for 37 % of all true positives and the *evaluation* criterion for 31 %, however in 38 % of all cases that *evaluation* was assigned any of the evaluation labels, 31 % were false positives, which is a fairly high amount even though the relative amount of true positives is 52 %.

Generally a considerable amount of false positives come from the fact that when a reviewer talks about some issue that falls under the *presentation* criterion, they refer to some section of the paper by the section name or topic, which often falls under an aspect expression belonging to some other criterion. Consider the sentence *"The section numbers appear off, too–the introduction says that evaluations are performed in Section 6, for instance, but it's actually in Section 5."*, which for a human reader obviously mentions a problem with the quality of writing, however the algorithm picks up on the *evaluations* keyword and incorrectly labels the sentence with the *evaluation* criterion.

Another issue that leads to a significant number of false positives is that at the beginning of most reviews, there is a summary of the paper, stating its objectives. These summaries often contain different aspect expressions, as there is an overlap between the criteria names or expressions and the general vocabulary of the field. For example, the sentence *"To achieve this, an inference rule is translated into a SPARQL CONSTRUCT query that is evaluated against the schema of the RDF dataset."* only mentions how the solution proposed in the reviewed paper works, it is not the reviewers comment on the quality of the actual evaluation

but it gets recognized by the algorithm as such. Similarly the sentence *"In this paper authors investigate how state-of-the art language technologies LT (tools, algorithms and resources) can be ported to the historical ecology domain."* gets labeled with the *state of the art* criterion, even though the sentence does not refer to how the reviewer feels about the research of the domain done by the authors (which would indeed fall under *state of the art*).

The false negatives, meaning cases where the annotators assigned a criterion to a comment but the algorithm did not, mostly come from the fact that the comments did not include any expressions that would immediately lead to a certain criterion. For instance in the case of the *technical quality* aspect it would require a significantly more complex domain specific lexicon of technical terms to lessen the amount of false negative in sentences such as *"Even if a rule is determined as potentially applicable after running the query derived from the rule on the data schema, the rule cannot be executed until relevant instances are entered into the dataset."*.

To explain the high amount of missing numerical values discovered in the previous section, the output of ISWC 2018 data was inspected and compared with the annotated dataset. It was discovered that out of the 7 "n/a" criteria values outputted for the 5 annotated reviews 4 criteria were also missing in the annotations (in other words according to the annotators there was no comment related to these criteria). All unknown values from the examined dataset belonged either under the *relevance* criterion (4 unknown values and 3 reviews with no *relevance* annotation) or the *technical quality* criterion (3 unknown values and 1 review with no *technical quality* annotation). As was already explained, the algorithm does not perform well in classifying comments regarding *technical quality* which is likely the cause of the missing values. On the other hand *relevance* has a significantly lower amount of false negatives, so it is more likely that reviewers simply do not feel the need to expand on their numerical score when this criterion is concerned.

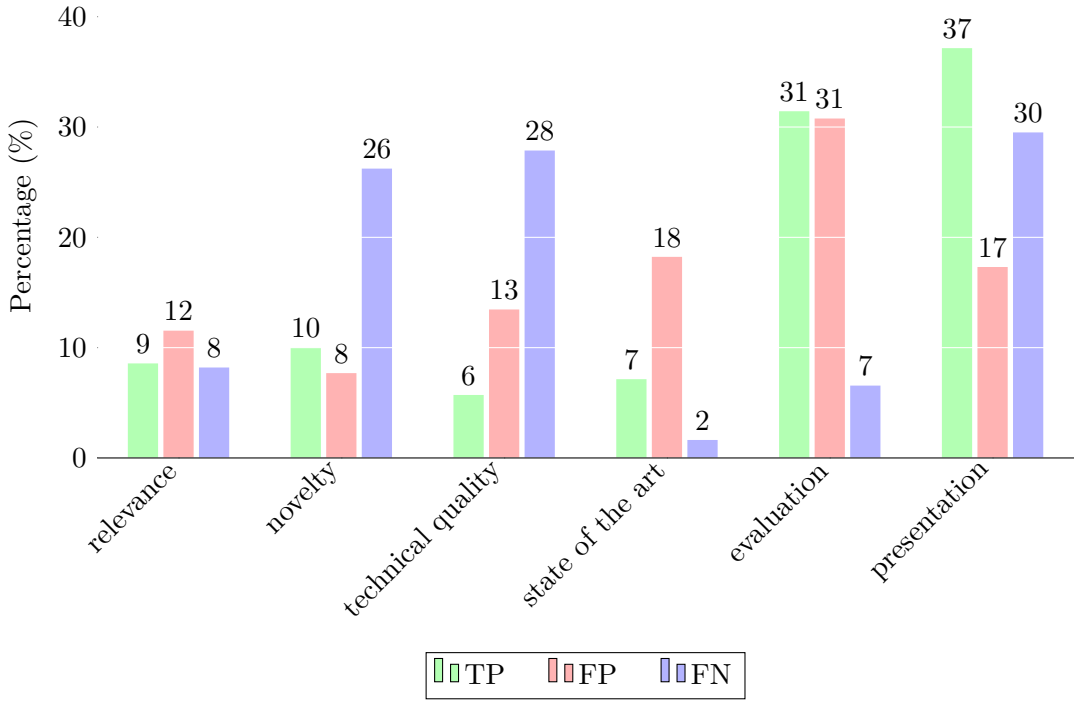

Figure 9.1: Evaluation results based on the contribution of each criterion to the evaluation labels

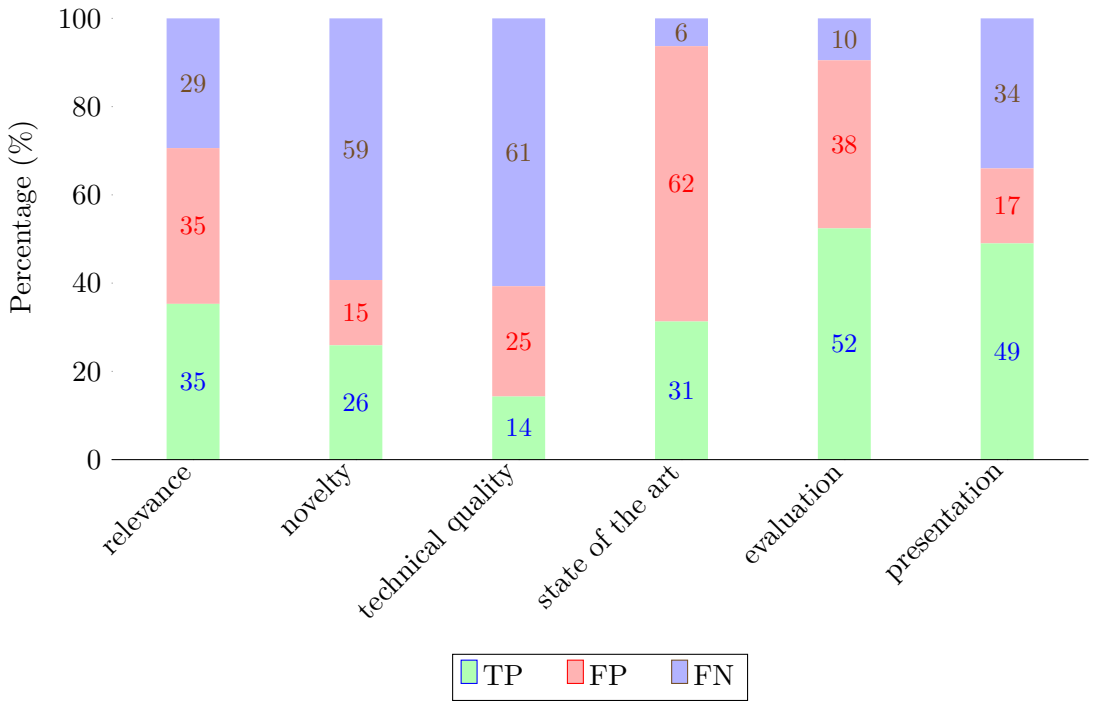

Figure 9.2: Evaluation relative to the number of assigned evaluation labels of each criterion

9.2.2 Sentiment analysis evaluation

The comments with aspects that were correctly classified (those with the TP label) were also evaluated based on whether they were labeled with the correct sentiment. In the annotated dataset there was only one comment where the annotators did not reach an agreement on the appropriate sentiment – *"Though, this approach solves a relevant problem there are several concerns:"*. This was due to that comment containing dual polarity which is an issue already pointed out in section 5.2.2. This comment could therefore be taken as either positive or negative and was evaluated accordingly.

Over 75.7 % of comments with correctly identified criterion were also correctly classified by sentiment while 24.3 % were not.

Relative to the number of TP comments regarding a certain criterion the algorithm performs best for the sentiment analysis of the *state of the art* criterion with an accuracy of 100 % (see Figure 9.3), however since only 5 comments belonging to this criterion were correctly identified, the number might not be objectively accurate. Interestingly the algorithm does well at determining the polarity of the *presentation* criterion as the comments tend to contain similar sentiment expression such as *well-written* or *clear* so due to their frequency across reviews they were picked up during the creation of the sentiment lexicon.

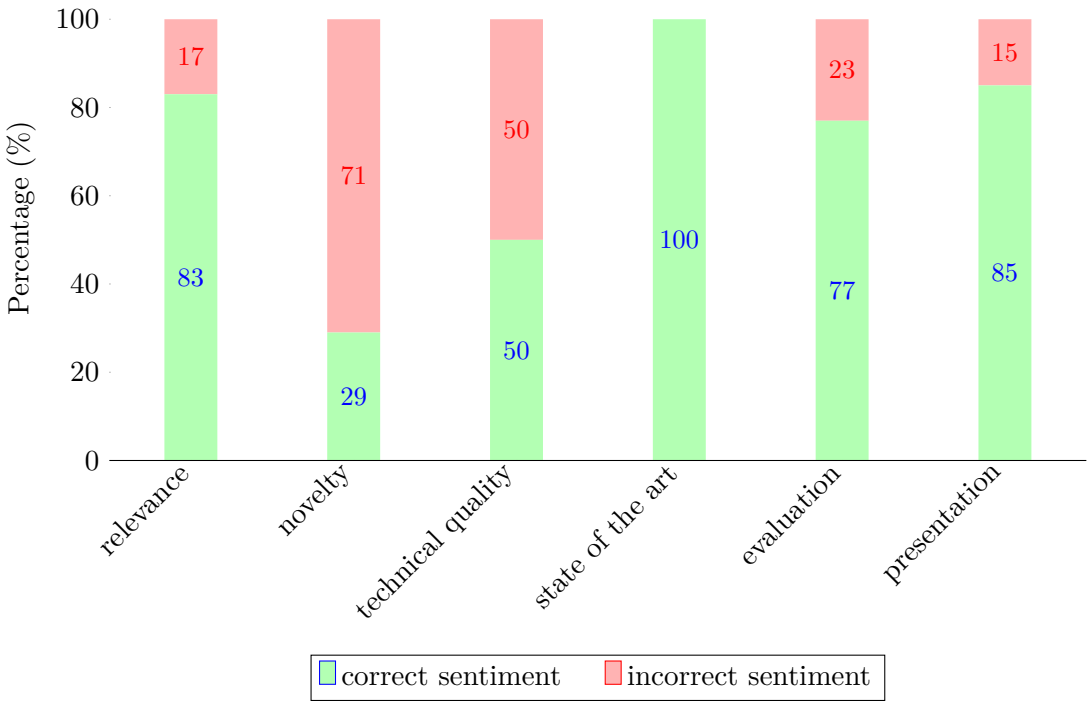

Figure 9.3: Results of sentiment analysis by criterion

9.3 Discussion of results

The comparison between the numerical output of the system with the numerical scores given in the reviews resulted in a mean average error of 0.99 on a scale from -2 to 2, meaning the algorithm was usually nearly one point off. This was deemed a significant error margin and in order to get more insight into the accuracy of the criterion identification and the sentiment analysis a more detailed assessment ensued on a sentence level which estimated the precision of the criterion identification at 57.38 % and the recall at 53.44 %. As such, the error rate is quite high, however even the annotators had a substantial level of disagreement, initially diverging in their criterion labeling in over a third of the comments. This suggests that reconstructing the intended meaning of the review comments is a particularly difficult task even for human annotators.

The accuracy of the sentiment analysis was calculated using a set of comments with correctly identified criterion determined by human reviewers. The system detected the accurate sentiment in more than 75 % of executed cases. Notably, this is an improvement on the result of a similar sentiment analysis carried out in a comparable study with focus on peer reviews of scientific publications. This analysis algorithm only managed to reach an accuracy level of nearly 73 % even though the sentiment analysis performed in this study was not aspect based [44], which makes determining the sentiment easier. Consider the sentence *"While it is a fairly relevant topic, there were too many typos and the results were not at all evaluated against any of the existing state-of-the-art systems, so I do not consider the work mature enough for acceptance."*. The overall sentiment of the sentence is negative, as it presents more reasons for rejecting the paper rather than accepting it and for that same reason it would not be difficult for most dictionary-based sentiment analysis methods to correctly classify the sentence as negative. However, if we classify the sentiment on an aspect level, it is necessary to also find out which sentiment expressions of the sentence belong to which aspect, which in this case means it needs to recognize that even though the sentence contains more negative expressions, relevance is judged positively.

The algorithm is capable of being substantially improved quite easily when given the availability of a significantly larger dataset. This could help put together both a better aspect expression dictionary and a vast sentiment lexicon. The creation of which heavily relies on the frequency of terms across reviews. Additionally, should a larger dataset be available, it would be possible to perform a more detailed exploration of the specificities of language used in similar data, creating a more complex algorithm using the discovered knowledge.

Another possible way to improve the results would be to put together specialized rules for the different criteria. For example, as comments about the *presentation* criterion often lead to false positives matches on other criteria, it would be possible to create a rule where criteria expression found in a sentence where there is also a match on *presentation* would be discarded. Another example of a criterion-specific rule would be to study the structure of a sentence in a more complex manner by using syntactic analysis, to for instance discover that in a sentence such as *"The rule cannot be executed until relevant instances are entered into the dataset."*

the adjective *relevant* is connected to *instances*, which is not a term related to the work itself and therefore the sentence should not be considered as a comment on the *relevance* criterion. However, these rules would require to limit the algorithm to a specific set of criteria, which goes against the original idea to create an algorithm that could be reused on a number of different domains.

The sentiment analysis as well as criterion identification could also be enhanced by creating a sentiment lexicon specific to each criterion. Firstly, there are many sentiment expressions that are context-dependent: For example, *small* would most likely considered a positive word in relation to an evaluation error but a negative one when commenting on the contribution of the work. This problem could be solved by having a dedicated sentiment lexicon for each criterion, and thus providing the necessary context. Secondly, some sentiment expressions have a strong connection to certain criteria. For example, *clear* is almost exclusively used in relation with the *presentation* criterion. This could be leveraged to aid in aspect identification, as a sentiment expression which is strongly related to a particular criterion could be used as an indication of its presence, if no criterion expression is found.

Conclusion

The aim of this thesis was to try to build a system for extracting opinions and sentiment from conference paper reviews. The objective was to determine if there is a way to successfully detect perceived value of a paper, based on sentiment analysis of its reviews. The results of this work show that the creation of an aspect-based sentiment analysis system focused on the domain of conference paper reviews is indeed possible.

In the practical part of this thesis a system was implemented, that analyzes a review following these steps: First an identification of which aspect of a paper each sentence of a review is commenting on. This is done by using a dictionary of aspect expressions, where each expression falls under one of the six chosen criteria – *relevance*, *novelty*, *technical quality*, *state of the art*, *presentation* and *evaluation*. Then it applies a dictionary-based method of sentiment analysis using a custom-built domain specific sentiment lexicon to determine the sentiment polarity of the criterion expression. By grouping the aspect expression polarities by the criteria to which they belong the system outputs numerical scores for each criterion on a scale from 1 to 5. It also lists all sentences of a review in which an aspect expression was identified, stating the respective criterion and its sentiment polarity in the sentence.

To evaluate the precision of such a system the numerical evaluation output of the reviews created by the algorithm was then compared to the original numerical scores from a set of reviews and the results were also evaluated in more detail by calculating the precision and recall of criterion identification on a sentence level. Based on the results of the evaluation a set of recommendations was given for future improvements, one of which being an acquirement of a significantly larger training dataset. Most conferences do not make their reviews publicly available and this was a considerable limitation of the implementation. In this aspect the secondary and unexpected objective of this thesis is to serve as a motivation for an easier accessibility of this kind of data.

The evaluation of sentiment analysis accuracy shows an improvement on the results of similar existing research and so with indicated improvements the system is going to be a valuable tool for helping to facilitate the meta-reviewing process. It can also help with the unification of criteria scores across different conferences and reviewers using the numerical scores outputted by the system.

All the algorithms created in this work – the aspect expression identification, sentiment lexicon compiler and aspect-based sentiment analysis – are implemented in a way which would require slight adjustments for application on a number of different domains of conferences. The main change that would be required is to manually create a new taxonomy of aspect expressions serving as a base for identification of other aspect expressions, helping to define the set of the domain-specific criteria. Allowing for minor adjustments, the practical use of the system in other domains is clear, providing the appropriate datasets to retrain the algorithm for the new application.

I am currently taking part in a project aimed at creating a generator of pictorial metaphors of reviews to simplify the difficult task of meta-reviewing by mapping the numerical scores of criteria to different parts of an image of a car [1]. The developed system has the potential to eventually work as an extension of this tool by allowing its use for reviews where the numerical scores are missing. This is just one of the many examples of future use of the system described in this thesis.

Should the reader of this thesis be inclined to find out more, the implementation is publicly available at `https://github.com/jurs02/aspect-based-sentiment-analysis-of-conference-submission-reviews`.

References

1. SVÁTEK, Vojtěch; JURANKOVÁ, Sára; ŠALDA, Radomír; STROSSA, Petr; VONDRA, Zdeněk. Creating and Exploiting the Mappings from Conference Review Forms to a Generic Set of Review Criteria. In: *Proceedings of the ISWC 2020 Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 19th International Semantic Web Conference (ISWC 2020)*. 2020. Available also from: `http://ceur-ws.org/Vol-2721/paper567.pdf`. online, accessed 2-November-2020.
2. LIU, Bing. *Sentiment analysis: mining opinions, sentiments, and emotions*. Cambridge University Press, 2015. ISBN 978-1-107-01789-4.
3. LIU, Bing. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data (Data-Centric Systems and Applications)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 3540378812.
4. SCHOUTEN, K.; FRASINCAR, F. Survey on Aspect-Level Sentiment Analysis. *IEEE Transactions on Knowledge and Data Engineering*. 2016, vol. 28, no. 3, pp. 813–830. ISSN 2326-3865. Available from DOI: `10.1109/TKDE.2015.2485209`.
5. HU, Minqing; LIU, Bing. Mining Opinion Features in Customer Reviews. In: *Proceedings of the 19th National Conference on Artifical Intelligence*. San Jose, California: AAAI Press, 2004, pp. 755–760. AAAI'04. ISBN 0-262-51183-5. Available also from: `http://dl.acm.org/citation.cfm?id=1597148.1597269`.
6. KUMAR, Ashish; SHARAN, Aditi. Deep Learning-Based Frameworks for Aspect-Based Sentiment Analysis. In: *Deep Learning-Based Approaches for Sentiment Analysis*. Ed. by AGARWAL, Basant; NAYAK, Richi; MITTAL, Namita; PATNAIK, Srikanta. Singapore: Springer Singapore, 2020, pp. 139–158. ISBN 978-981-15-1216-2. Available from DOI: `10.1007/978-981-15-1216-2_6`.
7. ZHANG, Lei; WANG, Shuai; LIU, Bing. Deep learning for sentiment analysis: A survey. *WIREs Data Mining and Knowledge Discovery*. 2018, vol. 8, no. 4, pp. e1253. Available from DOI: `https://doi.org/10.1002/widm.1253`.
8. WEBB, Geoffrey I. Naïve Bayes. In: *Encyclopedia of Machine Learning*. Ed. by SAMMUT, Claude; WEBB, Geoffrey I. Boston, MA: Springer US, 2010, pp. 713–714. ISBN 978-0-387-30164-8. Available from DOI: `10.1007/978-0-387-30164-8_576`.
9. JURAFSKY, Daniel; MARTIN, James. *Speech and Language Processing*. 3rd ed. draft. 2018. Available also from: `https://web.stanford.edu/~jurafsky/slp3/`.
10. PRINCETON UNIVERSITY. About WordNet. *WordNet A Lexical Database for English*. 2010. Available also from: `https://wordnet.princeton.edu/`.
11. RINKER, Tyler; SPINU, Vitalie. *sentimentr*. Zenodo, 2016. Version 0.4.0. Available from DOI: `10.5281/zenodo.222103`.

12. DING, Xiaowen; LIU, Bing; YU, Philip S. A Holistic Lexicon-based Approach to Opinion Mining. In: *Proceedings of the 2008 International Conference on Web Search and Data Mining*. Palo Alto, California, USA: ACM, 2008, pp. 231–240. WSDM '08. ISBN 978-1-59593-927-2. Available from DOI: 10.1145/1341531.1341561.

13. CARENINI, Giuseppe; NG, Raymond T.; ZWART, Ed. Extracting Knowledge from Evaluative Text. In: *Proceedings of the 3rd International Conference on Knowledge Capture*. Banff, Alberta, Canada: ACM, 2005, pp. 11–18. K-CAP '05. ISBN 1-59593-163-5. Available from DOI: 10.1145/1088622.1088626.

14. ASGHAR, Dr. Muhammad; KHAN, Aurangzeb; ZAHRA, Rabail; AHMAD, Shakeel; KUNDI, Fazal. Aspect-based opinion mining framework using heuristic patterns. *Cluster Computing*. 2019, vol. 22. Available from DOI: 10.1007/s10586-017-1096-9.

15. LYONS, John. Language, speech and writing. In: *Natural Language and Universal Grammar: Essays in Linguistic Theory*. Cambridge University Press, 1991, vol. 1, pp. 1–11. Available from DOI: 10.1017/CBO9781139165877.003.

16. GUDIVADA, Venkat N.; ARBABIFARD, Kamyar. Chapter 3 - Open-Source Libraries, Application Frameworks, and Workflow Systems for NLP. In: GUDIVADA, Venkat N.; RAO, C.R. (eds.). *Computational Analysis and Understanding of Natural Languages: Principles, Methods and Applications*. Elsevier, 2018, vol. 38, pp. 31–50. Handbook of Statistics. ISSN 0169-7161. Available from DOI: https://doi.org/10.1016/bs.host.2018.07.007.

17. YSE, Diego Lopez. *Your Guide to Natural Language Processing (NLP)*. 2019. Available also from: https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1. online, accessed 02-November-2020.

18. IORIN, Ilia. *Natural Language Processing (NLP) Use Cases for Business Optimization*. Available also from: https://mobidev.biz/blog/natural-language-processing-nlp-use-cases-business. online, accessed 02-November-2020.

19. TECHLABS, Maruti. *Top 12 Use Cases of Natural Language Processing in Healthcare*. Available also from: https://marutitech.com/use-cases-of-natural-language-processing-in-healthcare/. online, accessed 02-November-2020.

20. GOLDBERG, Yoav. Neural Network Methods for Natural Language Processing. *Synthesis Lectures on Human Language Technologies*. 2017, vol. 10, no. 1, pp. 1–309. Available from DOI: 10.2200/S00762ED1V01Y201703HLT037.

21. TURING, Alan M. Computing Machinery and Intelligence. *Mind*. 1950, vol. 59, no. October, pp. 433–460. Available from DOI: 10.1093/mind/LIX.236.433.

22. HUTCHINS, W. J. *Machine Translation: Past, Present, Future*. USA: John Wiley & Sons, Inc., 1986. ISBN 0470203137.

23. PAI, Aravind. *What is Tokenization in NLP? Here's All You Need To Know*. Available also from: https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/. online, accessed 02-November-2020.

24. MUJTABA, Hussain. *Tokenising into Words and Sentences | What is Tokenization and it's Definition?* 2020. Available also from: https://www.mygreatlearning.com/blog/tokenization/. online, accessed 03-November-2020.

25. CHAKRAVARTHY, Srinivas. *Tokenization for Natural Language Processing.* 2019. Available also from: https://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4. online, accessed 03-November-2020.

26. MANNING, Christopher D.; RAGHAVAN, Prabhakar; SCHÜTZE, Hinrich. *Introduction to Information Retrieval.* Cambridge University Press, 2008. Available from DOI: 10.1017/CBO9780511809071.

27. ENGINE, Sketch. *POS tags.* 2018. Available also from: https://www.sketchengine.eu/blog/pos-tags/. online, accessed 03-November-2020.

28. MARCUS, Mitchell P.; MARCINKIEWICZ, Mary Ann; SANTORINI, Beatrice. Building a Large Annotated Corpus of English: The Penn Treebank. *Comput. Linguist.* 1993, vol. 19, no. 2. ISSN 0891-2017.

29. ENCYCLOPAEDIA BRITANNICA, The Editors of. *Synthetic language.* Encyclopædia Britannica. Available also from: https://www.britannica.com/topic/synthetic-language. online, accessed 03-November-2020.

30. GARCÍA Clara, Cabanilles; PEDRO, Juan; RAMIREZ, Benjamín. *What is the difference between stemming and lemmatization?* Available also from: https://blog.bitext.com/what-is-the-difference-between-stemming-and-lemmatization/. online, accessed 03-November-2020.

31. BERI, Aditya. *Stemming vs Lemmatization.* 2014. Available also from: https://towardsdatascience.com/stemming-vs-lemmatization-2daddabcb221. online, accessed 03-November-2020.

32. ZOLA, Andrew. *The 5 Best Programming Languages for AI.* 2018. Available also from: https://www.springboard.com/blog/best-programming-language-for-ai/. online, accessed 15-November-2020.

33. BIRD, Steven; KLEIN, Ewan; LOPER, Edward. *Natural Language Processing with Python.* 1st. O'Reilly Media, Inc., 2009. ISBN 0596516495.

34. *WordNet with NLTK: Finding Synonyms for words in Python.* Guru99. Available also from: https://www.guru99.com/wordnet-nltk.html. online, accessed 09-November-2020.

35. PROJECT, NLTK. *nltk.corpus.reader package.* 2020. Available also from: http://www.nltk.org/api/nltk.corpus.reader.html?highlight=wordnet#nltk.corpus.reader.wordnet.Lemma.synset. online, accessed 09-November-2020.

36. *WordNet Interface.* Available also from: https://www.nltk.org/howto/wordnet.html. online, accessed 09-November-2020.

37. KAKARLA, Swaathi. *Natural Language Processing: NLTK Vs SpaCy.* ActiveState, 2019. Available also from: https://www.activestate.com/blog/natural-language-processing-nltk-vs-spacy/. online, accessed 09-November-2020.

38. *Mission*. Semantic Technology Institute International. Available also from: https://www.eswc-conferences.org/. online, accessed 03-November-2020.

39. *The Semantic Web: Research and Applications 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012, Proceedings*. 1st ed. 2012. 2012. Information Systems and Applications, incl. Internet/Web, and HCI ; 7295. ISBN 3-642-30284-X.

40. *About*. Knowledge Media Institute. Available also from: http://ekaw.org/. online, accessed 04-November-2020.

41. *22nd International Conference on Knowledge Engineering and Knowledge Management*. EKAW 2020. Available also from: https://ekaw2020.inf.unibz.it/. online, accessed 04-November-2020.

42. *International Semantic Web Conference (ISWC)*. SWSA (Semantic Web Science Association). Available also from: http://swsa.semanticweb.org/content/international-semantic-web-conference-iswc. online, accessed 04-November-2020.

43. SVÁTEK, Vojtěch; STROSSA, Petr. Let's Get the Best Papers to the Finish Line: Ontological and Pictorial Representation of Review Scores. 2019.

44. BUCUR, Cristina-Iulia; KUHN, Tobias; CEOLIN, Davide. *Peer Reviewing Revisited: Assessing Research with Interlinked Semantic Comments*. 2019. Available from arXiv: 1910.03218 [cs.DL].

45. TABOADA, Maite; BROOKE, Julian; TOFILOSKI, Milan; VOLL, Kimberly; STEDE, Manfred. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*. 2011, vol. 37, no. 2, pp. 267–307. Available from DOI: 10.1162/COLI_a_00049.

46. BUCUR, Cristina-Iulia; KUHN, Tobias; CEOLIN, Davide. *A Unified Nanopublication Model for Effective and User-Friendly Access to the Elements of Scientific Publishing*. 2020. Available from arXiv: 2006.06348 [cs.DL].

47. SABET, Peyman G. P.; ZHANG, Grace Q. "I don't think" versus "I think + not". *Text & Talk*. 01 May. 2017, vol. 37, no. 3, pp. 387–408. Available from DOI: https://doi.org/10.1515/text-2017-0010.

48. KARIMKHAN. *Extracting the noun phrases using nltk*. 2016. Available also from: https://gist.github.com/karimkhanp/4b7626a933759d0113d54b09acef24bf#file-noun_phrase_extractor-py-L34. online, accessed 30-October-2020.

49. MALOUF, Rob. *Source code for nltk.tokenize.mwe*. 2020. Available also from: https://www.nltk.org/_modules/nltk/tokenize/mwe.html. online, accessed 12-November-2020.

Attachments

A. Sentimentr study guide

Klasifikace sentimentu nástrojem *Sentimentr*

Z hlediska úlohy jde o standardní klasifikaci sentimentu jako pozitivního nebo negativního. Vhodné zejména pro dokumenty typu (produktových apod.) recenzí, kde lze za určitých okolností chápat původce a čas informací i hodnocený aspekt jako nezajímavé nebo implicitní hodnoty.

Přístup je založen na slovníkovém přístupu, přičemž je ale původní polarita pozitivních a negativních slov modifikována čtyřmi specifickými typy slov vyskytujícími se v jejich kontextu – jde o:

- negátory
- zesilovače ("amplifiers")
- zeslabovače ("de-amplifiers")
- odporovací spojky ("adversative conjuctions").

Zdrojové kódy i manuál jsou na `https://github.com/trinker/sentimentr`.

Sentiment věty a celého textu

Metoda se aplikuje na text složený z vět s_i, které jsou posloupnostmi slov w_{ij} (resp. výskytů slov, protože některá slova se mohou ve větě opakovat):

$$s_i = (w_{i1}, w_{i2}, ..., w_{in})$$

Dále se budeme zabývat jen aplikací metody na jednotlivou větu $s = (w_1, ..., w_n)$; index věty již neuvažujeme.

Metoda nalezne ve větě s všechny výskyty slov ze základního slovníku Pol obsahujícího polarizovaná slova. Jejich polarita $pol(w_i)$ může být nastavena buď na $+1$ vs. -1 (pokud slovník obsahuje jen prosté seznamy pozitivních a negativních slov), nebo pomocí specifických vah jednotlivých slov.

Sentiment věty $s = (w_1, ..., w_n)$ se pak počítá jako součet sentimentů všech výskytů slov ze slovníku Pol relativně k délce věty (resp. dle návrhu autora metody, k její odmocnině):

$$sentiment(s) = \frac{\sum_{w_i \in Pol} sentiment(w_i)}{\sqrt{|w_i|}}$$

Sentiment se počítá pro každou větu samostatně; agregaci na úroveň celého textu lze provést prostým průměrováním, případně jako vážený průměr, který může

např. s větší vahou započítávat záporné hodnoty (toto v systému zajišťuje volba `average_weighted_mixed_sentiment`).

Nyní tedy potřebujeme jen vědět, jak metoda vypočítá sentiment jednotlivých polarizovaných slov z věty, $sentiment(w_i)$.

Polarizované kontextové shluky

Původní polaritu každého polarizovaného slova w_i je nutno upravit na základě analýzy tzv. *polarizovaného kontextového shluku* $pcc(w_i)$. Výchozí podobou shluku pro dané polarizované slovo w_i (vzhledem ke kterému je kontext vytvářen) je posloupnost výskytů slov

$$pcc^{ini}(w_i) = (w_{i-b}, ..., w_{i-1}, w_i, w_{i+1}, ..., w_{i+a})$$

kde a a b jsou parametry označované jako `n.before` a `n.after`; jako jejich hodnoty jsou používány $a = 4$ a $b = 2$. Dále je aplikován slovník Pau obsahující "slova" vyjadřující "pauzu" (ve skutečnosti nemusí jít o slova, ale např. o znaky jako je středník): pokud je polarizované slovo od modifikátoru odděleno takovým slovem, modifikátor na něj nemá vliv. Formálně, každý výchozí shluk $pcc^{ini}(w_i)$ je upraven na redukovaný shluk takto:

$$pcc(w_i) = pcc^{ini}(w_i) \setminus \{w_j \mid \exists w_p \in Pau \text{ takové že } (j \leq p < i \ \vee j \geq p > i) \}$$

Aplikace modifikátorů

Jádrem metody je výpočet sentimentu jednotlivého slova, který vychází z jeho původní slovníkové polarity, ale upravuje ji na základě modifikátorů vyskytujících se v kontextovém shluku: negátory (resp. lichý počet negátorů) otáčejí polaritu, zatímco amplifikátory (amp), deamplifikátory ($deamp$) a odporovací spojky ($advcon$) modifikují intenzitu sentimentu:

$$sentiment(w_i) = pol(w_i) \cdot (-1)^{neg(w_i)} \cdot (1 + amp(w_i) + deamp(w_i)) \cdot advcon(w_i)$$

Pozn.: Některé části výpočtu nejsou přepsány zcela podle popisu na `https://github.com/trinker/sentimentr`*, ale intuitivním odhadem, protože v původním popisu buď nedávaly smysl nebo některá souvislost chyběla.*

1. U negátorů, ze slovníku Neg, se jednoduše zjistí jejich lichý nebo sudý počet.

   $$neg(w_i) = |w_j \in pcc(w_i) \cap Neg| \ mod \ 2$$

   Pozn.: Zápis množinové operace je zde, pro zjednodušení, vyjádřen matematicky nepřesně. $pcc(w_i)$ je sekvence s možností opakování (tj. multimnožina), zatímco Neg je neuspořádaná množina (bez opakování). Jako jejich průnik zde chápeme podsekvenci obsahující ty prvky z $pcc(w_i)$, které jsou i v množině Neg. Stejná konvence je i u ostatních slovníků/modifikátorů níže.

Dále, binarizace negátorového faktoru pomocí funkce zbytku (mod) není potřebná kvůli celkovému výpočtu z předchozího vzorce, ale kvůli použití tohoto faktoru v rámci de/amplifikace, viz níže.

2. Amplifikace se vyjádří jako:

$$amp(w_i) = (1 - neg(w_i)) \cdot ad \cdot |w_j \in pcc(w_i) \cap Amp|$$

kde *Amp* je slovník amplifikátorů a *ad* společná konstanta vyjadřující váhu amplifikátorů a deamplifikátorů, empiricky nastavená na 0,85. Pokud je ve shluku lichý počet negací, vliv amplifikátorů se v tomto vzorci "vypne" (resp. naopak se započítává níže jako deamplifikátor!), v opačném případě se každý amplifikátor započítává úměrně konstantě *ad*.

3. Deamplifikace se vyjádří jako:

$$deamp(w_i) = ad \cdot (|w_j \in pcc(w_i) \cap Deamp| + neg(w_i) \cdot |w_j \in pcc(w_i) \cap Amp|)$$

kde *Deamp* je slovník deamplifikátorů. Pokud je tedy ve shluku lichý počet negací, amplifikátory se započívávají tak, jako by byly deamplifikátory. Deamplifikace je opět úměrná konstantě *ad*.

4. Vliv odporovacích spojek je vyjádřen jako

$$advcon(w_i) = \\ 1 \; + \; ac \; \cdot \; (|w_j \in pcc(w_i) \cap AdvCon, j < i| \; - \; |w_j \in pcc(w_i) \cap AdvCon, j > i|)$$

kde *AdvCon* je slovník odporovacích spojek a *ac* je konstanta empiricky nastavená na 0,85. Spojka za polarizovaným slovem je tedy zeslabuje, spojka před polarizovaným slovem je naopak posiluje (intuice je, že "odporující" vyjádření chce pisatel oproti původnímu tvrzení zdůraznit).

Zpracoval V. Svátek, 2. 5. 2019