Discovery and Analysis of Public Opinions on Controversial Topics in the Educational Domain

Master-Thesis von Artem Vovk
31. October 2013

TECHNISCHE
UNIVERSITÄT
DARMSTADT

UBIQUITOUS
KNOWLEDGE
PROCESSING

Discovery and Analysis of Public Opinions on Controversial Topics in the Educational Domain

vorgelegte Master-Thesis von Artem Vovk

Supervisor: Prof. Dr. Iryna Gurevych
Coordinator: Oliver Ferschke

Tag der Einreichung:

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 31. Oktober 2013

(A. Vovk)

Zusammenfassung

Argumentation ist wichtig für Menschen im täglichen Leben als auch bei der Arbeit. Damit ist verbunden Argumente für oder gegen ein bestimmtes Thema darzulegen, um anhand dessen Informationen zu präsentieren oder eine Entscheidung zu treffen. Der Bildungsbereich dient hier als gutes Beispiel. Bachelorabsolventen stehen oft vor der Wahl, ob sie das Masterstudium fortsetzen wollen oder eine berufliche Karriere anfangen. Um eine solide Entscheidung treffen zu können, ist es für sie grundlegend die Vor- und Nachteile der beiden Möglichkeiten zu kennen. Das Web ist überfüllt mit Daten deren Umfang stetig wächst. Mit einer großen Anzahl an Argumenten für viele verschiedene Themen, stellt das Internet eine große Hilfe dar. Ein Problem besteht jedoch darin die relevanten Argumente zu finden. Die gängigen Suchmaschinen können diese Aufgabe nicht erfüllen, weshalb man nach immer intelligenteren Lösungen für das Problem sucht. An genau dieser Stelle kommt "Argumentation Mining" ins Spiel.

In dieser Thesis präsentieren wir einen konzeptuellen Entwurf von einem System, dessen Aufgabe darin besteht, Argumente zu einem bestimmten Thema zu finden. Wir schlagen vor dieses System als eine Suchmaschine zu implementieren, die nach der Eingabe einer Suchanfrage zu einem bestimmten Thema, nach passenden Argumenten sucht. Aufgrund beschränkter Rechenleistung, begrenzen wir uns nur auf in Deutsch verfassten Texten aus der Bildungsdomäne. Zudem implementieren und evaluieren wir die wichtigsten Teile des Systems wie: den Crawler, das Argumentenextraktion- und Klassifikationsmodul und die Frontend-Schnittstelle. Für das Extraktions- und Klassifikationsmodul verwenden wir Techniken des überwachten maschinellen Lernens. Der Prozess beginnt mit dem Sammeln von Dokumenten, welche relevante Argumente enthalten. Im nächsten Schritt definieren wir ein Annotationsschema und führen eine Annotationsstudie durch. Als Ergebnis erstellen wir einen beschrifteten Korpus, welcher benutzt wird, um Versuchsmodelle für Argumentenextraktion und -klassifikation zu trainieren. Zusätzlich evaluieren wir den Einfluss von verschiedenen Klassifikationsalgorithmen auf das System. Zum Schluss untersuchen wir die Auswirkung von verschiedenen Merkmalkombinationen und führen eine Fehleranalyse durch.

Abstract

Argumentation is used by everybody in their daily lives as well as work. People frequently need to identify arguments in favor or against a specific topic in order to present some information or make a decision. The educational domain serves as good example. Bachelor graduates often find themselves wondering if they should pursue a Master's degree or start working in the industry. Finding pros and cons of each possibility is crucial for them in order to make up their mind. The Web is overloaded with data and it is growing constantly. It includes many arguments for topics in various fields but people are not satisfied anymore with traditional search engines that are supposed to find these arguments. Therefore, they look for more intelligent solutions and this is where argumentation mining comes in play.

In this work we present a conceptual design of a system with the task to simplify the access to argumentation information concerning a specific topic. We propose to implement such a system as a search engine which looks for the arguments in the Web given a topic as a query. Because of the computation limitations we decide to concentrate only on topics from the educational domain and arguments in german language. We also implement and evaluate the critical parts of the system such as: a focused crawler, argument extraction and classification module as well as the front-end interface. For the extraction and classification part we decide to use supervised machine learning techniques. Therefore, first we collect the documents which contain the arguments. Secondly, we define the annotation scheme and perform the annotation study. As a result we create a labeled corpus, which is used for training models for the argument extraction and classification experiments. Finally, we evaluate the influence of different classification algorithms as well as the combination of different features and perform the error analysis.

Contents

1 Introduction

This chapter describes the motivation and provides the objectives and the structure of this thesis.

1.1 Motivation

Argumentation is an inherent aspect of almost every proficiency field. In order to defend some idea or come up with a plausible conclusion, many professionals such as lawyers, scientists and journalists need to submerse themselves with the advantages and drawbacks of the concerning topics. This requires searching for documents, articles and books that contain arguments in favor or against their topic of interest. An example from the juristic domain could be a lawyer trying to convince the court about her client's innocence. This requires a good understanding of the law itself and finding law statements in favor of the defended client. In this thesis, from all the possible topic domains, we choose to focus on the educational domain. The system of education seems to be a controversial field for the majority of the population. Many students for example have a dilemma after graduating with their Bachelor's degree. They have to make up their mind whether they wish to pursue a Master's degree or move on to the industry. In order to regret their final decision as little as possible, they need to consider strong arguments in favor, as well as against the possible options.

Nowadays, the Web serves as a huge argumentation pool, combining information from various scientific disciplines and incorporating all possible information sources. Processing this information with the aim to find and analyze arguments regarding the topics of interest would be of much benefit, for example for the mentioned students concerned about their future study choices.

Unfortunately, the extraction and analysis of argumentation structures from the Web requires complicated intellectual input from the user due to the natural limitations of the human processing capacity. The search for relevant texts by entering queries in a search engine provides answers that are not necessarily what the user was expecting. The search engine does not take into account whether the found documents actually correlate with the user's objective or whether they even include arguments concerning the topic the user was interested in. An example could be a query *"Master's degree pros and cons"*. The search engine might provide documents about Master studies in different universities, completely ignoring that the user was searching for arguments or find texts that actually include the arguments, but again, the user would have to search for them in the text on her own. In the face of a constantly growing corpus of information, this makes the use of traditional search engines tiresome and ineffective and forces the field of artificial intelligence to offer automated solutions. This is where argumentation mining with the aim to retrieve and analyze arguments, comes into the picture.

1.2 Objectives

The main goal of this work is to design and develop a prototype of a system which task is to simplify the access to argumentation information with a particular focus on the educational domain. The prototype should be able to find arguments in favor or against a specific educational topic and present the results in a structured way. The target language for this system is German, however it should be designed to be as language independent as possible. The developed prototype should have three main components: data extraction, argument extraction and classification and presentation component. The first component should be implemented as a focused crawler, which uses a model learned from educational texts. The second, and the most important, is the argument extraction and classification component. In order to fulfill its task it is designated to use supervised machine learning techniques. However, this requires availability of labeled corpus, which should also be created. The last component is the web-based user interface, which should be able to present the extracted and classified arguments from the previous component.

1.3 Structure of the thesis

The thesis begins with the introduction part (Chapter 1) that describes its motivation as well as the objectives, which should be considered.

Since, to the best of our knowledge, there are no existing publicly available similar solutions of our system, we compare only the components of our system. This is done in Chapter 2.

In Chapter 3 we introduce the conceptual design of traditional search engines and also describe a design of our proposed solution.

Chapter 4 presents a short introduction to the argumentation theory and additionally provides a definition of the argument in our context.

From Chapter 5 we start to describe the components we develop. In Chapter 5 we introduce the focused crawler used for data retrieval of educational texts from the Web.

In Chapter 6 we describe the process of the corpus creation. This corpus is further used for the supervised machine learning experiments of the argument extraction and classification component. This component is presented in Chapter 7 and is considered to be a key component of the system.

In Chapter 8 we introduce a prototype of a user interface of the proposed search engine with argument information.

The work that has been done throughout this thesis as well as the introduction of possible future work in this field is summarized in the last chapter.

2 Related work

To the best of our knowledge there is no existing publicly available solution, which is similar to our work. However, this year, German Research Center for Artificial Intelligence (DFKI) announced a start of a project referred to as ARGUMENTUM[1]. This system should provide innovative methods for computer-assisted analysis, retrieval and synthesis of argumentation structures in the legal domain. One of the main goals of this project is to create a prototype for performing search for user entered legal questions and present arguments regarding these questions. This means they have a similar goal to our system, but in a different domain. They have also already published several theoretical works about this system [HFL$^+$12] [HNFL13]. However, the planned release date is no sooner than in 2015.

There are not many related works in the context of argumentation mining, which is one of the most important aspects of our prototype. In their work [MBPR07] Palau and Moens et al. conduct experiments regarding automatic detection of arguments in legal texts (English texts). They consider this task a classification task. For this purpose they also apply supervised machine learning algorithms, which are trained on a set of annotated arguments. This set is constructed from he structured data in the Araucaria [RR04] corpus. They conduct several experiments with different feature sets by using only two classifiers: multinomial Naive Bayes and the maximum entropy classifier. The features include n-grams, adverbs, verbs, modal auxiliaries, different text statistics, punctuation, word couples, depth of the parse tree as well as specific key words. However, the features they used only provided 0.69% (accuracy) of improvement in comparison to basic n-grams. In our experiment we outperform our n-gram baseline by 4.1% (accuracy). Moreover, they do not provide any classification scheme for arguments.

In the further work of Palau and Moens [PM09] the authors focus more deeply on the argumentation theory and perform experiments regarding the argument structure. They consider an argument as a combination of premises and conclusions and conduct corresponding experiments. Our work does not provide coverage for argument structure, however, we propose it as part of future work which can be performed on our system.

Another work regarding argumentation mining, and also one of the first to appear, is Argumentative zoning [Teu99]. The main goal of this study is to analyze the argumentative status of sentences in scientific papers. For this purpose the authors manually create and annotate a corpus, which consists of 203 academic papers. They also define an annotation scheme for classifying the argumentative zone and distinguish seven of them. The automatic classification of argumentative zones is based on supervised machine learning (Naive Bayes, Maximum Entropy, RIPPER) and sentential features. However, the annotation and developed features are very dataset specific and cannot be applied to different kind of texts.

We also find several works regarding polarity classification for opinion mining. In our prototype we use this to determine the polarity of a particular argument (classification of `Arguments by Polarity` in Section 7.5.4). One of the most recent and interesting works in this field is [WK09]. The authors of this work apply supervised machine learning techniques for polarity determination, just like we do. They use a subset of the popular MPQA[2] corpus in order to train a model. This corpus contains manually annotated news articles in English language. For classifying the authors use only support vector machine. They also define a set of useful features, some of them are reused in our work (e.g. polarity of different part of speeches, occurrence of polarity changers). However, with the best classification configuration they are able to outperform the n-gram classification performance by 8.9% (average F1-measure). In our experiments we perform better than the n-gram baseline by 10% (average F1-measure). Moreover,

[1] http://www.dfki.de/web/presse/pressemitteilungen_intern/2013/saarbrucker-forscher-entwickeln-suchmaschine-fur-argumentationen

[2] http://mpqa.cs.pitt.edu/

they only determine the polarity of sentences and do not consider the target at all, our problem is more complicated, since we determine polarity of a sentence or group of sentences towards an argument target.

3 Conceptual design

In this chapter, we introduce the conceptual design of common search engines and describe our proposed solution.

3.1 General solution

Search engine is one of the most popular technologies in the field of information retrieval. The goal of every search engine is to obtain a list of relevant documents found in the Web to a given query provided by a user. Nowadays, there exist a lot of academic works and industrial solutions in this field. We analyze some of these works ([BP98], [ZQDS03], [HGS10]) in order to define the general high-level architecture, which can be reused for our solution.

Figure 3.1 shows the resulting architecture of a general search engine. It should have a crawler, a storage, an information retrieval component and the index. The crawler is used to retrieve the information from the Web and save it in the systems storage. In its turn, storage, should be accessible by the information retrieval component, which task is to extract the relevant information from crawled pages (e.g. title, URLs, plain text large font etc. [BP98]). Afterwards the whole processed and extracted information should be stored in the efficient index.

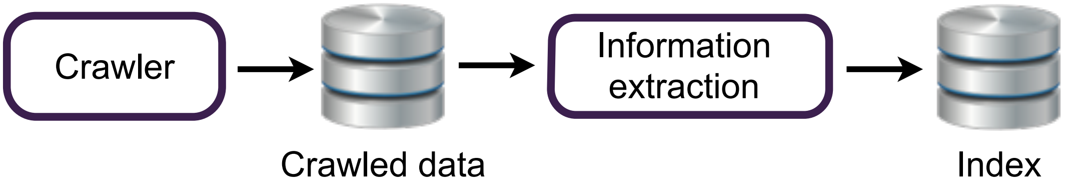


Figure 3.1: Concept architecture of a search engine

3.2 Proposed design

On the basis of the general architecture of search engines described in the previous section we designed our own solution. Figure 3.2 presents the conceptual design of the search engine system we propose. First, as every search engine, it needs a crawler for exploring the Web. Since we decide to concentrate only on the educational domain we need a special crawler (marked as 1), which looks only for specific pages. Such a crawler is a called focused crawler. In order to estimate the relevance of visited Web pages to a given topic in a classical focused crawler, we need to have a model (marked as 2) related to the target domain [BPM09]. Implementation of the crawler and its model is described in Chapter 5.

After the crawler performs its job it should store the crawling results (marked as 3), which are later used by the argument extraction and classification component (marked as 4). The task of this component is to examine the acquired data for argument occurrence and classify detected arguments (Chapter 7) according to the scheme presented in Section 6.3. For this purpose we propose to use machine learning classification techniques, which requires a labeled corpus (marked as 5, Chapter 6). As a result this component should produce a list of extracted arguments and their classification. Furthermore, this should be stored in the index (marked as 6), which in its turn, make the information accessible for the front-end (marked as 7, Chapter8).

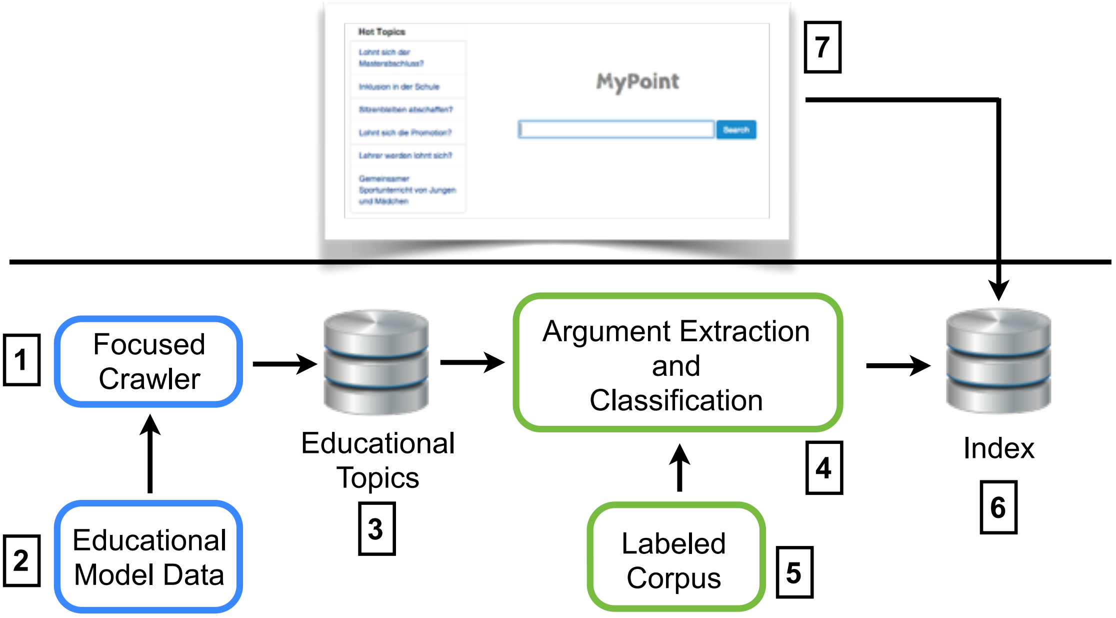

Hot Topics
Lohnt sich der Masterabschluss?
Inklusion in der Schule
Sitzenbleiben abschaffen?
Lohnt sich die Promotion?
Lehrer werden lohnt sich?
Gemeinsamer Sportunterricht von Jungen und Mädchen
MyPoint
Search
7
1
Focused Crawler
Educational Topics
3
Argument Extraction and Classification
4
Index
6
2
Educational Model Data
Labeled Corpus
5

Figure 3.2: Conceptual diagram of the proposed system

In this work we concentrate on the development of the components for such a system. In particular we implement the focused crawler, argument extraction and classification module as well as the corresponding labeled corpus and front-end.

4 Argumentation theory

Argumentation is a crucial aspect in many scientific disciplines as well as in everyday life. An example could be a court trial where a lawyer can only support his client well by providing strong arguments in his favor. A more familiar example, targeting a large group of people, could be finding arguments connected to educational topics which are often controversial. Many Bachelor students, for instance, have to decide whether or not to pursue a Master's degree later on. Finding strong arguments is crucial for them. This is exactly what argumentation mining is needed for. Argumentation mining deals with automatic detection of argumentation structures in a document and combines natural language processing, argumentation theory and information retrieval. However, the definition of an argument and argumentation itself is more controversial that it might appear. According to [PM09] there are three main argumentation theories which have practical meaning for argumentation mining. All three of them have one thing in common: they all agree that an argument, the unit of argumentation, is formed by premises and a conclusion. The first one focuses on assigning predefined meanings to parts of text according to their role within the argument. Unfortunately applying this theory in practice is troublesome due to complex reasoning structures in free texts. The second theory understands argumentation as a dialog between person in favor and a person against a specific topic, where the protagonist tries to convince the antagonist of her point of view. Last but not least, the most common theory defines argumentation schemes. This is also the theory on which the definition of an argument formed by [PM09] is based on: "An argument is set of propositions, being all of them premises, except maximum one, which is a conclusion." A proposition is denoted by a declarative sentence or sometimes a smaller text span used to make a statement or assertion. In this thesis, we do not restrict to this classical definition (which requires the presence of premises and a conclusion) and present a more relaxed notion of an argument. We include so-called "enthymemes“". These are arguments in which one or more propositions that are part of the argument are missing [WRM08]. In other words the argument is formed only by premises or a conclusion. Consider the following example: "Teachers do not earn a lot of money". This is an enthymeme since it is an argument missing a conclusion. A completion of it could look as follows: "Teachers do not earn a lot of money, making this profession unrewarding".

Since we propose a search engine that is meant to display the arguments to people, the conclusion of such a statement a statement could be easily inferred by the user. She can easily see that this is a fact that does not speak in favor of becoming a teacher and can be therefore assumed to be an opposing argument for the topic "Becoming a teacher pays off". Due to this human supervision of the search engine results, we take such "incomplete" arguments into account as well.

5 Crawler

In this chapter, we present a focused crawler developed for the extraction of pages from the educational domain. This crawler is based on the simplified version of the approach introduced by Kumar and Vig in their work [KV13].

5.1 Background

Crawler is one of the most important parts of every search engine. The main task of a crawler is the retrieval of documents available in the Web by traversing them from one link to another. In contrast to classical crawler, the focused crawler is designed to collect Web documents which are relevant to a specific domain and tries to avoid the irrelevant ones. This decreases the overall load of the network and consumption of computation resources [KV13].

A common focused crawler usually needs a set of initial URLs (seeds) and a model which is used for the retrieval of relevant pages. Such a model should contain a set of pages from the target domain. In this way the topic relevance of the crawled pages may be computed as a similarity of those pages to the existing model [HMY+11]. In order to compute the similarity measure we need a model for representing text documents as well as a similarity algorithm. The most common and easy to implement model is the vector space model (VSM). According to VSM, each document is represented as a vector:

$$d_j = (w_{1,j}, w_{2,j}, ..., w_{t,j}) \tag{5.1}$$

Here t is the number of terms in all documents and $w_{t,j}$ is the weight of term t in the document d_j. There are several different approaches to compute the term weights. One of the most common and well-known is the Tf-Idf (Term frequency-Inverse document frequency) [HMY+11]. This measure denotes the word importance for a document in a collection, and is computed as:

$$w_{t,d} = log(1 + tf_{t,d}) * idf_t \tag{5.2}$$

Here $tf_{t,d}$ is the term frequency of term t in document d: number of times that term occurs in a document. The idf_t is the inverted document frequency, which is computed as:

$$idf_t = log(N/df_t) \tag{5.3}$$

where N is a number of documents in the collection and df_t (document frequency) is the amount of documents that contain term t.

In the VSM the similarity between two documents usually depends on the distance between vectors. The most common measure for computing the similarity between vectors is the cosine similarity, shown below:

$$cosSim(d,q) = \frac{\sum_t w_{td} w_{tq}}{\sqrt{\sum_t w_{td}^2 \sum_t w_{tq}^2}} \tag{5.4}$$

In the next section we describe how we applied the above mentioned theory to our crawler.

5.2 Developed crawler

As a basis for our crawler we decide to use a Python framework called Scrapy[1]. It is a Python application framework for Web crawling and information extraction. It provides a user friendly application programming interface (API) and includes a lot of functionality. In a simple case you just need to provide Scrapy with the list of seed URLs and define what kind of information you want to extract and the rest of the work is done for you. An overview of some of the Scrapy features[2]:

- Built-in support of selecting and extracting data from HTML
- Built-in support for cleaning and sanitizing the scraped data using a collection of reusable filters
- Built-in support for generating feed exports in multiple formats (JSON, CSV, XML)
- Support for extending Scrapy by plugging your own functionality using signals and a well-defined API (middleware, extensions and pipelines)

However, Scrapy framework does not have direct support of focused crawling which needs to be implemented. In order to solve this problem we write our own spider which examines the content of each crawled page and computes the similarity between the page and the model (train set). As a train set we take the content of twenty pages related to the educational domain on the topics: *"Sitzenbleiben"*, *"Lehrerberuf"*, *"Promotion"* and *"Masterstudium"*. After that we preprocess this data by using the following operations:

1. Remove stop words from each page of the train set.
2. Apply stemming to each page from the train set.
3. Compute Tf-Idf weight for each term of each document.
4. Compute a mean vector of all Tf-Idf vectors.

For the first and second step we use the Python Natural Language Toolkit (NLTK[3]), which is a leading platform for building Python application with natural language processing capabilities. We compute Tf-Idf weight by using Equation 5.2. In order to find a mean vector we use averaging: add the respective weights and divide by the number of vectors. This mean vector is used then for the similarity estimation by computing cosine similarity (Equation 5.4) between mean and the Tf-Idf vector of a crawled page.

As seed URLs for our crawler we use the educational categories of the most popular german newspaper and magazine websites, such as: `www.spiegel.de`, `www.sueddeutsche.de`, `www.welt.de`, `www.zeit.de` etc. The full list of seed URLs can be found in Appendix A.

After we prepare our training set for the similarity computation and determine the set of seed URLs, we implement the crawling algorithm (see Algorithm 1). According to this algorithm, first we need to create a priority queue and add all seed URLs with the maximum priority to it (line 1 - 3). This ensures that seed URLs are extracted before other URLs. After that we start to iterate over priorityQueue (line 4 - 15). We dequeue URL from this queue, download page corresponding to this URL, extract its content, compute Tf-Idf score, determine the similarity of the content to the mean vector of the training set and finally save this data (line 5 - 9). Furthermore, we obtain all links from the downloaded page and calculate for each link its total score by adding the similarity score of the page to the similarity score of link's anchor text. Then we enqueue these links with their scores to the crawler priority queue (line 11 - 14).

The entire process is repeated until the crawl queue is empty or it is stopped by a third party.

[1] http://scrapy.org/
[2] https://media.readthedocs.org/pdf/scrapy/0.18/scrapy.pdf
[3] http://nltk.org/

Algorithm 1: Focused crawling

```
// Initialize priority queue with max priority for seed URLs
foreach seedURL do
    enqueue($crawlPriorityQueue, seedURL, MAX\_PRIORITY$);
end
// Perform crawling
while crawlPriorityQueue is not empty do
    $URL$ = dequeue($crawlPriorityQueue$);
    $pageContent$ = extractContent($URL$);
    $pageTfIdf$ = computeTfIdfScore($pageContent$);
    $similarityScore$ = cosineSimilarity($meanVector, pageTfIdf$);
    savePage($URL, pageContent, similarityScore$);
    foreach link in page do
        $linkTfIdf$ = computeTfIdfScore($link.anchorText$);
        $linkSimialrityScore$ = cosineSimilarity($meanVector, linkTfIdf$);
        $totalScore = similarityScore + linkSimialrityScore$;
        enqueue($crawlPriorityQueue, link, totalScore$);
    end
end
```

5.3 Evaluation

Since the main focus of this thesis is not a crawler, we decide to use sanity testing as fast evaluation method. Sanity test is a simple check to see if the obtained results are suitable or not. For this purpose we select top 50 documents from the crawled set according to similarity and manually check their relevance to the educational domain. This shows that 40 pages (80%) are relevant and 10 not. Among the irrelevant pages 8 contain just a list of all topics for a particular date. A lot of those topics are education related, however this is evaluated as a false positive extraction since we are only interested in pages which content is about the education, and not just list of short topics. Other two pages contain material about psychological experiment regarding the power of words and about the retirement in Germany.

6 Corpus creation

In this chapter, we describe the process of the corpus creation, which includes: creation of the annotation scheme, selection of corpus topics and documents, corpus annotation and corpus evaluation.

6.1 Motivation

Since we decide to use supervised machine learning techniques for the argument segmentation and classification, it is necessary to have a gold standard (labeled corpus). As it is mentioned above, our system is focused on the educational domain and German language. Unfortunately, there is no freely accessible corpus with these peculiarities in the Web. For this reason we decide to create a new one.

6.2 Selection of corpus topics and documents

As a part of the task description we are provided with about fifteen different topics in the educational domain. We plan to use ten of them in the final corpus. In order to retrieve sufficient amount of data for the given topics, we manually analyze more than 1000 pages in the Web. To retrieve those pages we use our crawler (described in Chapter 5) and the top 100 Google search results, with a topic as a query. The selection criterion of the pages is the occurrence of at least three arguments in the text of the page. This analysis shows that many of the given topics do not have enough controversial pages about them in the Web. Only four ("Sitzenbleiben abschaffen?", "Inklusion in der Schule", "Lehrer werden lohnt sich?", "G8 oder G9") of fifteen originally given topics are taken to the end corpus. By the further analysis of educational domain we find other three topics ("Lohnt sich der Masterabschluss?", "Lohnt sich die Promotion?", "Sportunterricht: Jungen und Mädchen zusammen?"). We are left with the total of seven topics . The detailed corpus statistics is depicted in Table 6.1.

Topic	Amount of pages
Sitzenbleiben abschaffen?	22
Lehrer werden lohnt sich?	19
Lohnt sich die Promotion?	13
G8 oder G9?	12
Inklusion in der Schule	8
Sportunterricht: Jungen und Mädchen zusammen?	8
Lohnt sich der Masterabschluss?	7
Total	89

Table 6.1: Corpus statistics

The documents used in the corpus are articles from German newspapers and magazines (e.g. `http://www.spiegel.de/`, `http://www.sueddeutsche.de/`, `http://www.zeit.de/`, `http://www.focus.de/` etc.). We choose them because of the good quality and structure of the text and the availability of constructive arguments. The full list of URLs can be found in Appendix B.

6.3 Annotation scheme

For the structured and organized annotation process it is indispensable to have a clear annotation scheme. First, we describe what is an argument (Chapter 4) and define the elementary units of annotation. By analyzing the corpus documents we decide to use *sentences* as elementary units of an argument, the reasons for that are:

- we do not find sentences which contain multiple arguments
- easy to annotate for humans
- easy to parse for sentence tokenizers
- used in the similar works [PM09], [Teu99]

Then we introduce three different argument classification types:

- Arguments by Polarity (Pro/Contra).
- Arguments by Argumentative Type (Qualitative/Quantitative).
- Arguments by Reference (Referenced/Unreferenced).

In the following we describe these classification types.

6.3.1 Arguments by Polarity

According to this classification we consider all arguments either to be supporting (Pro) for a given controversial topic or opposing (Contra). This type of classification mainly comes from the definition of the argument and is widely used in different works [PM09], [HNFL13]. It is important to mention that these labels are strongly dependent on the topic name. It means that for the same text the arguments are different if we negate the topic of the text, e.g. topic *"Mac is better than Windows"* has reversed Pro/Contra labels as the same text but with the topic *"Mac is worse than Windows"*.

The description of the classification with the examples from the corpus is shown in Table 6.2.

Label	**Description**	**Example (topic: "Sitzenbleiben abschaffen")**
`Pro`	Argument which supports given controversial topic	*Besonders junge Menschen sind dagegen, weil sie eine sinkende Leistungsbereitschaft fürchten.*
`Contra`	Argument which opposes given controversial topic	*Es sei kontraproduktiv, wenn ein Schüler ein ganzes Jahr wiederholen müsse, obwohl er vielleicht nur in einem bestimmten Fach Defizite habe, sagt etwa die grüne Bildungsexpertin Ina Korter.*

Table 6.2: Arguments by Polarity

6.3.2 Arguments by Argumentative Type

This type of classification is created by analyzing text arguments from the educational domain. We observed that some of the arguments contain statistical data, results of surveys and polls as well as other important numbers for the argument polarity. Here is an example of such an argument: *"PhDs earned 21,50€ per hour in their first year, while MSc. graduates only earned 17,50€"*. In this sentence we have direct income comparison because of using specific values. We call these arguments as quantitative arguments. Other arguments, which are just based on general statements are referred to as qualitative arguments.

Table 6.3 presents the short description of this classification as well as the examples from the corpus.

Label	**Description**	**Example (topic: "Lohnt sich die Promotion?")**
`Quantitative`	Argument which is based on statistical data as a result of surveys, studies or polls as well as argument with numbers that are important for the argument polarity	*Laut der Vergütungsstudie der Unternehmensberatung Kienbaum über Leitende Angestellte bekommt ein Universitätsabsolvent mit Promotion 144.000 Euro im Jahr, ohne Doktor nur 131.000 Euro.*
`Qualitative`	Argument which is based on general statements	*Objektiv lohnt sich die Promotion: Mit einem Doktortitel winken in der freien Wirtschaft bessere Aufstiegschancen, mehr gesellschaftliches Ansehen und höhere Gehälter.*

Table 6.3: Arguments by Argumentative Type

6.3.3 Arguments by Reference

This type of classification is based on the analysis of arguments from the created corpus. We notice that some of the arguments have explicit references (citations) which describe their origination and some not. This motivates us to distinguish between these arguments. For example: "According to the DummyOrg study, 90% of MSc. graduates get higher income than BSc. graduates". This sentence includes the direct source of the argument: "DummyOrg study".

Table 6.4 shows the description and examples of classification Arguments by Reference.

Label	**Description**	**Example (topic: "Lohnt sich der Lehrerberuf?")**
`Referenced`	The source of the argument is given or it is directly based on the opinions of other entities	*Auch im Vergleich mit anderen und ähnlich belasteten Berufen wie Polizisten, Pflegern, Beschäftigten im Strafvollzug und im Sozialbereich zeigten sich bei Lehrern generell die ungünstigsten Konstellationen, sagte Schaarschmidt.*
`Unreferenced`	The source of an argument is unknown or unclear	*Doch Lehrer haben kaum Aufstiegschancen - und die Fleißigen werden kaum belohnt.*

Table 6.4: Arguments by Reference

6.4 Annotation process

The annotation is performed by three annotators from the same social background. We divide the annotation task into two steps:

- Detection of argument boundaries (limits of the argument).
- Classification of selected arguments according to three defined classes.

It is important to mention that for the annotation task we use all the documents described in Section 6.2. However, the documents with the topic "G8 oder G9?" are considered twice during annotation, once

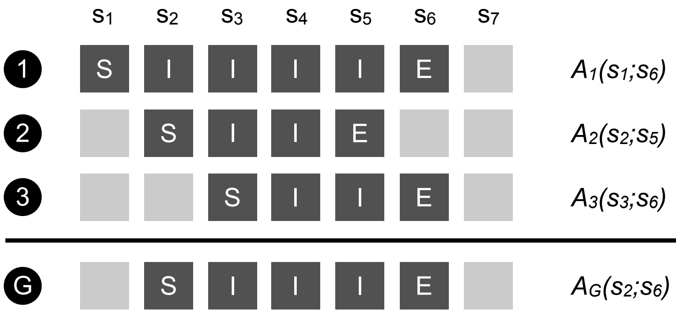


Figure 6.1: Example of the automatic boundary reconciliation algorithm

with the title “G8” and once with “G9”. The reason for that is the peculiarity of the developed system: detection of arguments in favor or against a specific topic, and not a comparison between different topics, which is a more challenging task.

In the first step the annotators detect arguments in the document by selecting the argument sentences. After all three annotators have processed the document we execute an algorithm for the automatic argument boundary reconciliation (described in Section 6.4.1). This algorithm processes the annotator’s boundaries and creates the “gold” boundaries for the document. After this processing annotators start with the second step, where they should classify the “gold” boundaries according to the three different classification criteria described in previous section.

Before we start with the annotation, we instruct the annotators and perform a pilot study in order to identify possible problems. The results of this study show that the annotators have a significant disagreement about the argument boundaries and difficulties with the classification by Argumentative Type. We discuss with the annotators the problems mentioned above. After that we perform the final study, which takes about two weeks.

6.4.1 Algorithm for the automatic boundary reconciliation

To the best of our knowledge there is no existing algorithm for boundary reconciliation. Therefore, we decide to implement a new one, which works automatically. As a basis for this algorithm we take the majority voting principle, which is frequently used for label reconciliation (e.g. [AB13]). Figure 6.1 illustrates an example of this algorithm. As an input we have the annotated data from three annotators and the goal is to find the best match between them. First, we label each sentence (s_n) for each annotator (A_n), according to the following scheme:

- S - first argument sentence.
- I - intermediate argument sentence.
- E - last argument sentence.
- O - one sentence argument (which consists from one sentence).

After this we go through all sentences and perform majority voting for them (in case of three annotators majority is at least two votes). For example, s_1 has only one S label, and since it is not enough we move further on to s_2. s_2 has two labels (S and I), therefore this sentence should be a part of an argument. In order to determine which sentence is it, we now perform majority voting between these labels: $N(S) = N(I) = 1$. It is a draw, and for these cases we use the special rules. These rules are application dependent, in our case, with an equal amount of S labels and I labels we choose S as a resulting label. We proceed till the end of the document analogously.

The pseudocode of this approach is described in Algorithm 2.

Algorithm 2: Automatic boundary reconciliation

```
input  : Annotated document by $n$ annotators
output: Gold boundaries in the document

// Label document sentences
foreach annotator do
    foreach annotated sentence do
        label sentence with ($S$, $I$, $E$, $O$);
    end
end

// Perform majority voting
foreach sentence do
    $labels$ = get sentence labels;
    if $labels$ size > $n/2$ then
        $probable\_labels$ = get most frequent $labels$;
        if probable_labels size > 1 then
            apply special rules for choosing most frequent label;
        end
        add most frequent label to the result;
    end
end
```

6.5 Annotation tool

In order to increase the annotator's performance and motivation we decide to implement a web-based annotation tool with a user friendly interface. Figure 6.2 shows the user interface (UI) of the designed tool. It consists of four main regions (marked in Figure 6.2). The first one shows all enumerated documents which are used in the annotation. The color of each document number represents the current processing status and has the following meaning:

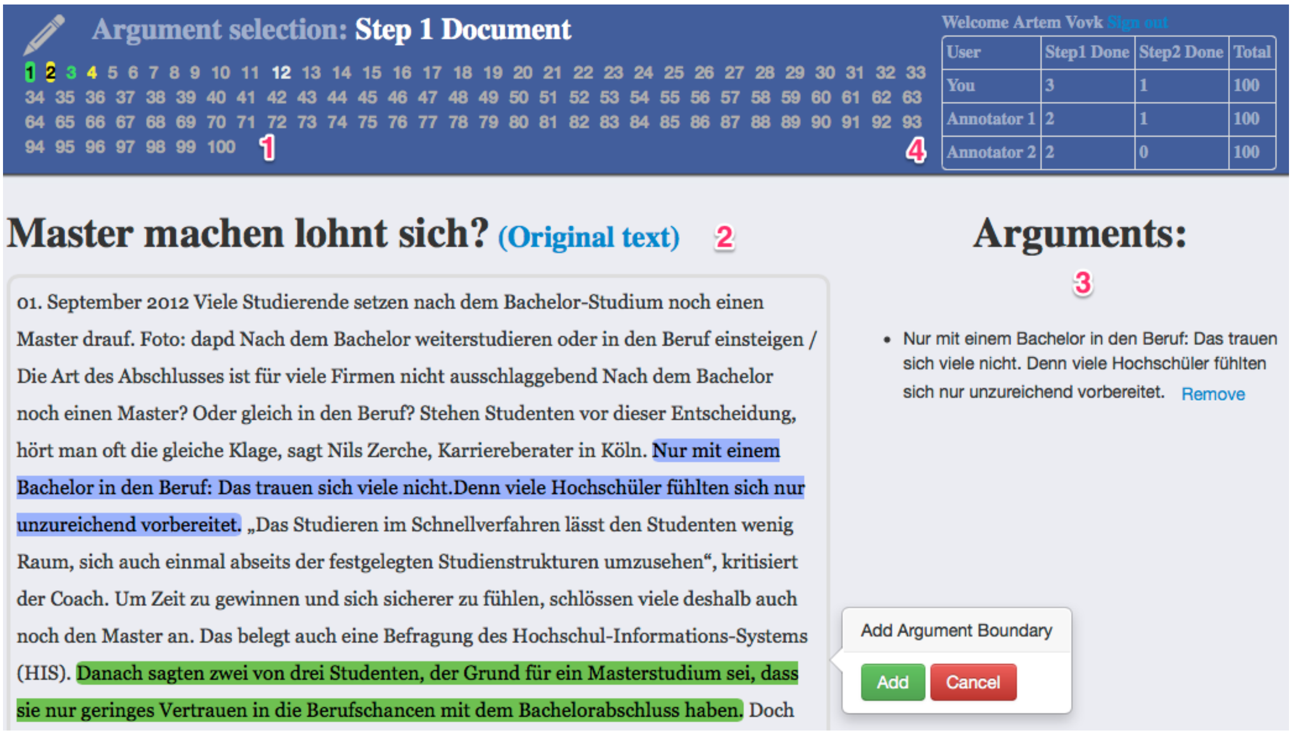
- 1 *Label 1* (white color) - Currently selected document
- 1 *Label 2* (gray color) Unprocessed step 1 document
- 1 *Label 3* (yellow color) Processed but not approved step 1 document
- 1 *Label 4* (green color) Approved step 1 document
- 1 *Label 5* (black color with yellow background) Unprocessed step 2 document (if three annotators approve the same document it will be automatically changed to this color)
- 1 *Label 6* (black color with green background) Processed step 2 document (completely processed document)

The second region of the UI includes a title of the selected document, a text and a link to the original page. The third region shows currently selected arguments of a text and the fourth region provides a user with the processing status of the annotation.

Depending on the label we differentiate different types of documents. Labels 1-4 means that the document is currently step 1 document. Only after all annotators approved the same document (Label 4) the system automatically calculates the gold boundaries (see Section 6.4.1) and converts this document to step 2 document (Label 5).

Figure 6.2 shows the system with the step 1 document. Here a user can only choose the argument boundaries and either just "Annotate" it or "Approve" it (Figure 6.3). "Annotate" means that the document is only partially processed or needs a review later. "Approve" means that the annotator is confident in his decision about the argument boundaries and from this time on the document can be automatically converted to the step 2 document (if all annotators approve it).

In Figure 6.4 a step 2 document is depicted. In step 2 document annotators see a text and the selected arguments (region 1). Each of these arguments should be classified according to the three classifications or marked as "Not an Argument" (region 2). After all the arguments in the text are classified by an annotator the document is marked with Label 6, which means that it's processing is completed.



Argument selection: Step 1 Document

Welcome Artem Vovk Sign out

User	Step1 Done	Step2 Done	Total
You	3	1	100
Annotator 1	2	1	100
Annotator 2	2	0	100

Master machen lohnt sich? (Original text)

01. September 2012 Viele Studierende setzen nach dem Bachelor-Studium noch einen Master drauf. Foto: dapd Nach dem Bachelor weiterstudieren oder in den Beruf einsteigen / Die Art des Abschlusses ist für viele Firmen nicht ausschlaggebend Nach dem Bachelor noch einen Master? Oder gleich in den Beruf? Stehen Studenten vor dieser Entscheidung, hört man oft die gleiche Klage, sagt Nils Zerche, Karriereberater in Köln. Nur mit einem Bachelor in den Beruf: Das trauen sich viele nicht.Denn viele Hochschüler fühlten sich nur unzureichend vorbereitet. „Das Studieren im Schnellverfahren lässt den Studenten wenig Raum, sich auch einmal abseits der festgelegten Studienstrukturen umzusehen", kritisiert der Coach. Um Zeit zu gewinnen und sich sicherer zu fühlen, schlössen viele deshalb auch noch den Master an. Das belegt auch eine Befragung des Hochschul-Informations-Systems (HIS). Danach sagten zwei von drei Studenten, der Grund für ein Masterstudium sei, dass sie nur geringes Vertrauen in die Berufschancen mit dem Bachelorabschluss haben. Doch

Arguments:

- Nur mit einem Bachelor in den Beruf: Das trauen sich viele nicht. Denn viele Hochschüler fühlten sich nur unzureichend vorbereitet. Remove

Figure 6.2: Annotation Step 1: argument boundary detection

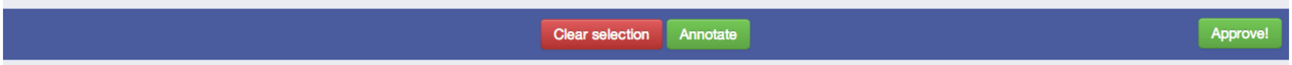
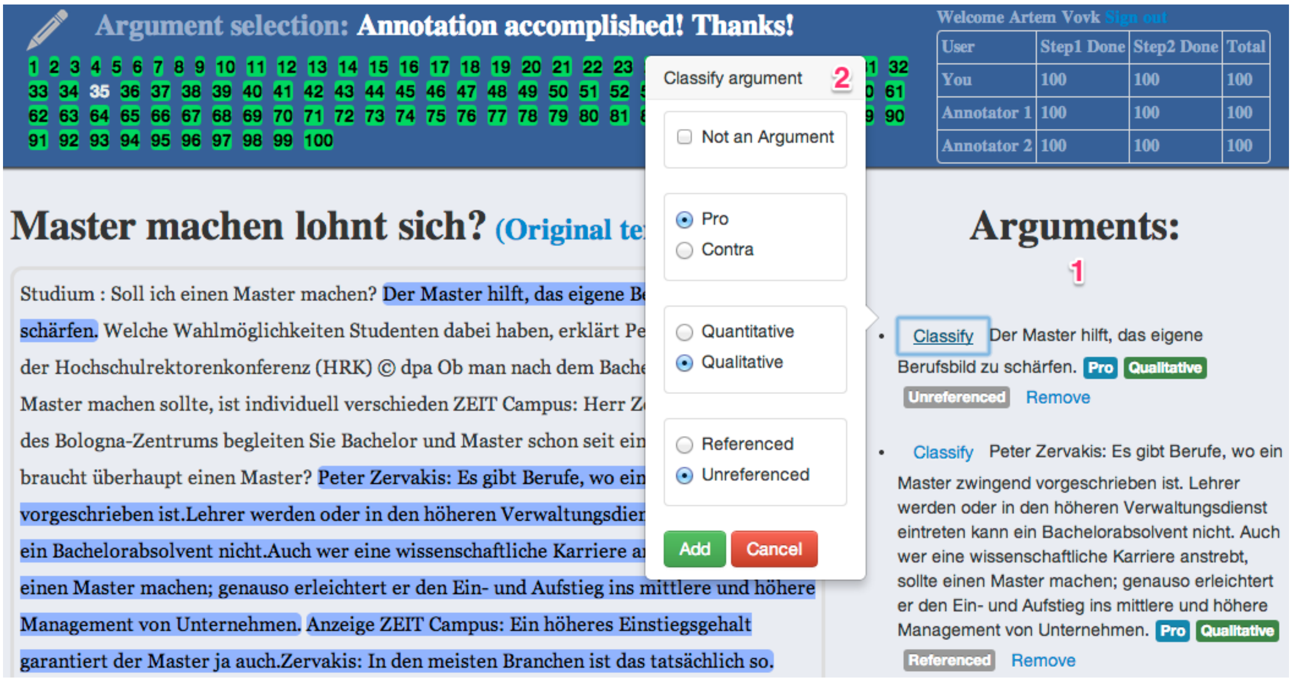


Figure 6.3: Bottom part of the annotation tool

6.6 Annotation evaluation

In this section we present the evaluation of inter-annotator agreement for argument boundaries as well as the inter-annotator agreement for three argument classification types defined in the annotation scheme. Furthermore, we also evaluate the algorithm for the automatic boundary reconciliation by using the simple error metric.



Argument selection: Annotation accomplished! Thanks!

Welcome Artem Vovk Sign out

User	Step1 Done	Step2 Done	Total
You	100	100	100
Annotator 1	100	100	100
Annotator 2	100	100	100

Master machen lohnt sich? (Original te

Studium : Soll ich einen Master machen? Der Master hilft, das eigene Be
schärfen. Welche Wahlmöglichkeiten Studenten dabei haben, erklärt Pe
der Hochschulrektorenkonferenz (HRK) © dpa Ob man nach dem Bache
Master machen sollte, ist individuell verschieden ZEIT Campus: Herr Z
des Bologna-Zentrums begleiten Sie Bachelor und Master schon seit ein
braucht überhaupt einen Master? Peter Zervakis: Es gibt Berufe, wo ein
vorgeschrieben ist.Lehrer werden oder in den höheren Verwaltungsdien
ein Bachelorabsolvent nicht.Auch wer eine wissenschaftliche Karriere an
einen Master machen; genauso erleichtert er den Ein- und Aufstieg ins mittlere und höhere
Management von Unternehmen. Anzeige ZEIT Campus: Ein höheres Einstiegsgehalt
garantiert der Master ja auch.Zervakis: In den meisten Branchen ist das tatsächlich so.

Classify argument

☐ Not an Argument

● Pro
○ Contra

○ Quantitative
● Qualitative

○ Referenced
● Unreferenced

Arguments:

- Classify Der Master hilft, das eigene Berufsbild zu schärfen. Pro Qualitative Unreferenced Remove
- Classify Peter Zervakis: Es gibt Berufe, wo ein Master zwingend vorgeschrieben ist. Lehrer werden oder in den höheren Verwaltungsdienst eintreten kann ein Bachelorabsolvent nicht. Auch wer eine wissenschaftliche Karriere anstrebt, sollte einen Master machen; genauso erleichtert er den Ein- und Aufstieg ins mittlere und höhere Management von Unternehmen. Pro Qualitative Referenced Remove

Figure 6.4: Annotation Step 2: argument classification

6.6.1 Inter-annotator agreement for argument boundaries

Computation of inter-annotator agreement for argument boundaries (segmentation task) is not as straightforward as it might look. The classical evaluation approaches like generalized Cohen's κ [DF82] or Scott's π [Fle71] tend to be very low, since annotators generally agree on the availability of segments, but they disagree on their exact boundaries [AP08]. To demonstrate such a case, consider the following example; two coders annotate 47 sentences and put three boundaries each. They agree on two boundaries, but disagree only on one sentence for the third boundary. In this situation the κ coefficient is 0.65, which is pretty low considering that the difference is only one sentence. Therefore, we decide to use more sophisticated approaches.

To overcome the problems mentioned above we use the following metrics: average pairwise argument overlap and a new state of the art metric - boundary similarity (described in the next subsection) [Fou13]. The first metric provides us with general confidence that the annotation process was feasible by calculating the argument overlaps, and the second incorporates the principle of near misses and shows actual the agreement.

To compute pairwise argument overlap we use the following method. First we take a pair of annotators separately. Then for each argument of this pair we give a score of one if one of the argument sentences overlaps with another argument sentence and a score of zero if not (Figure Figure 6.5). We sum up these scores for each pair and normalize them by dividing by the amount of all arguments of each of the annotators. Then we compute the average value between all pairs of annotators. This gives us a value of **0.84**. It means that only 16% of all detected arguments do not overlap. We consider these results to indicate the annotators at least understood what sentences can form an argument.

The second metric can compute the inter-annotator agreement only for different types of boundaries. Therefore, we consider our task as annotation of two types of boundaries: argument begin boundary and argument end boundary. For calculation of boundary similarity we use a tool, called Segeval[1], and obtain a value of **0.36** (scale from 0 - disagreement to 1 - full agreement). Since [Fou13] does not specify

[1] SegEval tool can be found at `https://segeval.readthedocs.org/en/latest/`

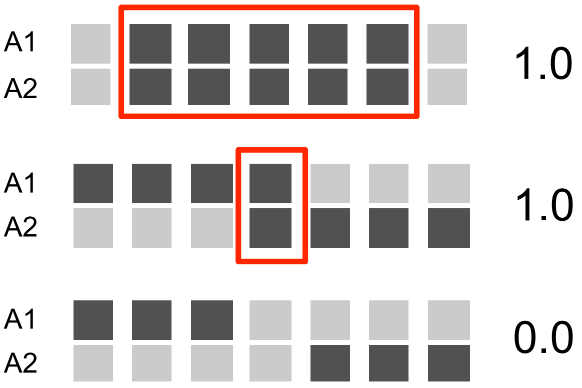


Figure 6.5: Average pairwise argument overlap

the interpretation scale for boundary similarity, we evaluate it by using following methods: comparison with random annotators and comparison with other corpora.

To compare with the random annotators performance we first compute the probability distribution of putting begin/end argument boundary for each annotator separately. Then by using this probability distributions we generate three annotations for each of the texts in corpus. We compute the inter-annotator agreement for these three random annotators and get the value of 0.11. This value significantly differs (by 69%) from the actual agreement and this means that annotators did not perform at random.

Since, to the best of our knowledge, there is no free publicly available annotated (with the annotations from different users) corpus of arguments, we compare our agreement to other data sets: The Stargazer data set [Hea97] and The Moonstone data set [KS12]. They significantly differ from our corpus: topical segmentation on the paragraph level and with only one boundary type (we have two boundary types) [Fou13], but it gives us a rough idea about the variation of the agreement in segmentation tasks. The Stargazer data set gives us an agreement of 0.44, while The Moonstone only 0.28[2]. Agreement of our dataset is directly between those two agreements. Table 6.5 summarizes the boundary similarity values of different annotated corpora.

Annotated corpus	**Inter-annotator agreement (boundary similarity)**
Created corpus (real annotators)	0.36
Created corpus (random annotators)	0.11
The Stargazer data set	0.44
The Moonstone data set	0.28

Table 6.5: Boundary similarity values of different annotated corpora

Boundary similarity

In his work [Fou13], Fournier proposed a new metric for inter-coder agreement in segmentation tasks with the property to award partial credit for near misses. This metric is called boundary similarity. It uses three main edit operations to model segmentation comparison:

[2] Average value for each group of 4-6 coders [Fou13]

- Additions/deletions when full miss occurs (AD).
- Substitutions if one boundary placed instead of another (S).
- n-wise transpositions for near misses (T).

Figure 6.6 shows an example segmentation of two texts t_1 and t_2 and boundary edit operations applied on them. Here we have one near miss T (for the miss distance 2), a matching pair of boundaries M as well as two full missed AD. Furthermore, for each of this operations a correctness score is assigned. The mean value of this score is used as normalization of boundary edit distance. Further information about the score values and normalization computation can be found in [Fou13].

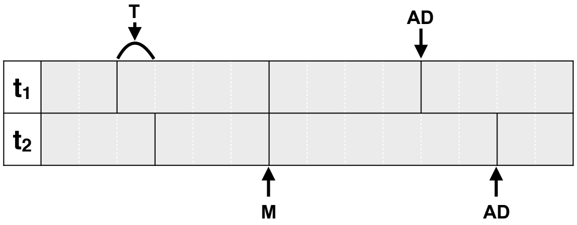


Figure 6.6: Boundary edit operations

6.6.2 Inter-annotator agreement for argument classifications

In order to compute the inter-annotator agreement for each of the three classification types we choose the chance corrected κ (Kappa) [Coh60] coefficient, in particular its generalized version for multiple coders - multi-κ [AP08]. This can be computed by using the following formula:

$$\kappa = \frac{A_o - A_e}{1 - A_e} \tag{6.1}$$

A_o here refers to the so called observed agreement - the proportion of items on which the annotators agree. A_e is the agreement expected by chance, which is calculated based on individual annotator label assignment distribution. The ratio between $A_o - A_e$ and $1 - A_e$ gives us the actual agreement beyond chance. The detailed computation of A_o and A_e values for multi-κ is described in [AP08].

Table 6.6 shows inter-annotator agreement for three classification types.

Classification type	Observed agreement	Expected agreement	Kappa
Arguments by Polarity	0.96	0.5	0.93
Arguments by Argumentative Type	0.92	0.68	0.75
Arguments by Reference	0.88	0.5	0.77

Table 6.6: Inter-annotator agreement for three classification types

We interpret these Kappa values by using the benchmark scale of [LK77] (Table 6.7). According to this scale we get an almost perfect agreement for Arguments by Polarity and substantial agreement for two others. In the following, we analyze the most common sources of annotator's disagreement.

Kappa value	Agreement
< 0.0	Poor
0.0 - 0.20	Slight
0.21 - 0.40	Fair
0.41 - 0.60	Moderate
0.61 - 0.80	Substantial
0.81 - 1.00	Almost perfect

Table 6.7: Interpretation of Kappa. Scale of [LK77]

For the classification of Arguments by Polarity the main source of errors is the presence of multiple negative sentiments or negations. For example, if the topic has a negative sentiment *“Sitzenbleiben abschaffen”* and the argument also has negative sentiment *“Sie halten das Sitzenbleiben mehrheitlich für schädlich und demotivierend...”*. Such cases are causing a lot of confusion and increase a cognitive load [WvL08] of the annotators which results in higher error rate.

For the Arguments by Reference the most popular non-agreeing case is when the argument is a part of a long citation. For example, the argument *“Ohnehin ist sich der engagierte Schulleiter sicher, dass vor die Wahl gestellt, die Schulkonferenz sich für G9 entscheiden würde. Denn die Mehrheit der Eltern steht dem Leistungsprinzip immer noch skeptisch gegenüber.”* does not explicitly state the source. However, only the next sentence has the source *“...fürchtet Salbrecht.”*.

Another frequent case where the annotators disagree on the referenced label is the not clear description of the source. Consider the following arguments:

- ***Eltern****: Auch die Elternbeiräte im Land wollen zurück zu G9 ...*
- ***Forscher der Hochschule*** *hatten 112 Lehrer aller Schularten in Baden-Württemberg befragt. Fast zwei Drittel der Lehrer schätzten das Ansehen ihres Berufsstandes als m̈angelhaftëin ...*
- *... so zeigen* **die Hamburger Zahlen***: Acht Jahre Gymnasium können ausreichen.*

These sources (“Eltern”, “Hamburger Zahlen”, “Forscher der Hochschule”) are actually not clearly defined and this makes the classification of such arguments complicated for the annotators.

In the classification of Arguments by Argumentative Type we notice that one of the annotators labeled only the quantitative arguments which are based on the results of studies and polls and did not take into consideration arguments with important sentiment numbers (e.g. *“Promovierte der Uni Köln verdienen ein bis zwei Jahre nach ihrem Abschluss durchschnittlich 21,21 Euro brutto pro Stunde, während Diplom-, Magister- und Masterabsolventen nur 17,90 Euro verdienen.”*). Main reason for this is that the pilot study did not contain the arguments based on numbers, but only the ones with results of studies and polls. Therefore, we missed it in the error analysis performed after the pilot study. However, other two annotators captured these cases and it did not affect the quality of created corpus, only the agreement.

6.6.3 Evaluation of algorithm for the automatic boundary reconciliation

We also evaluate our algorithm described in Section 6.4.1. For this purpose we use simple error metric:

$$e = \frac{N_d - N_a}{N_d} = \frac{592 - 572}{592} = 0.034 \tag{6.2}$$

Where N_d is the amount of arguments detected by our algorithm - 592 and N_a the actual amount of arguments - 571. The first value we get after annotation step 1. The second, since in step 2 the annotators reviewed all the extracted arguments and either classified it (in this case they are agree that this is an argument) or marked it as not an argument, we calculate as an amount of arguments classified by the majority as an argument.

According to this value only 3.4% of all arguments detected by our algorithm are rejected by human annotators. We consider this value to be acceptable, since we lost only a small amount of all arguments.

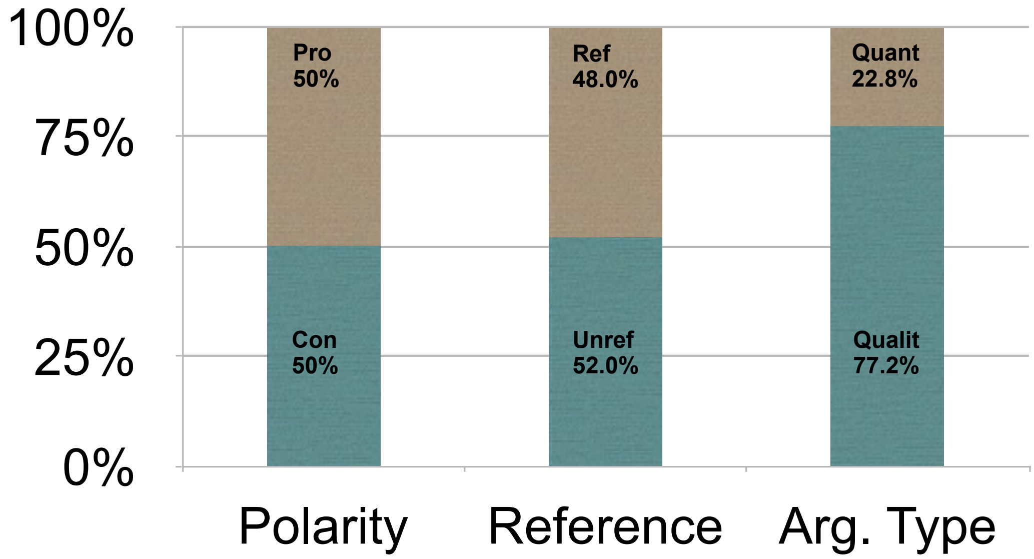


Figure 6.7: Label distribution in corpus for each of three classification types

6.7 Created corpus

In this section we present and analyze properties of the created dataset.

As we mentioned in the previous sections after step 1 we had about 592. Twenty of these arguments were rejected by the majority (two or more annotators). However, for our experiments we decide to take into account only the arguments on which all three annotators agreed. The reason for this is that we want to include only high-quality arguments in order to avoid possible noise, which is crucial for many classification algorithms [NOPF10] .Therefore, in the end we come up with 487 arguments.

Furthermore, we calculate the label distribution for each of three classification types (Figure 6.7). As we can see, there is balanced distribution in polarity and reference classification type, but unbalanced in Argumentative Type. We have four time more qualitative arguments than quantitative. This peculiarity should be considered in the classification experiments.

Statistic	Min	Max	Avg.
Argument length (sentences)	1	8	2.12
Amount of arguments per text	0	14	4.92
Text length (sentences)	11	222	44.85

Table 6.8: Simple text and argument statistic

Moreover, we calculate the amount of arguments as well as the label distribution for each of the topics presented in the corpus. It is shown in Table 6.9.

Table 6.8 presents other important statistic of the created dataset. First is the argument length, which is important for argument extraction. As we can see an average argument in our corpus consists of about two sentences. Afterwards we also calculate, the average amount of arguments per text and get the value of five. Here it is important to mention that one of the texts does not contain arguments at all. By analyzing this case we find out that even though each of the annotators detect one argument in this text, they disagree on the argument boundaries. The first and the third annotator do not have any intersection of argument boundaries at all. The argument of the second annotator overlaps with the argument of the first annotator by one sentence, however this sentence does not build a complete argument on its own.

Topic	**Polarity**		**Argumentative Type**		**Reference**		**Total args.**
	Pro	*Contra*	*Qualit.*	*Quantit.*	*Ref.*	*Unref.*	
Sitzenbleiben abschaffen	55	56	71	40	78	33	111
Lohnt sich die Promotion?	46	37	74	9	29	54	83
Lehrer werden lohnt sich?	22	41	41	22	24	39	63
G8	11	46	42	15	25	32	57
Inklusion in der Schule	24	26	45	5	19	31	50
G9	40	8	37	11	21	27	48
Lohnt sich der Masterabschluss?	30	9	30	9	23	16	39
Sportunterricht: Jungen und Mädchen zusammen?	14	22	36	0	15	21	36

Table 6.9: Label distribution for each of the topics presented in the corpus

7 Classification experiments

In the chapter we present an argument extraction and classification component. The goal of this component is to extract with maximum performance. In order to compare the performance, first we established a baseline as well as the upper bound. We also compare the results of some experiments to existing similar works.

7.1 Frameworks used for argument classification and extraction

For the argument classification and extraction task we decide to use DKPro Text Classification[1] (DKPro TC) framework. This is a UIMA-based[2] text classification framework, which incorporates the DKPro Core[3] and DKPro Lab[4] frameworks as well as the well-known Weka Machine Learning Toolkit[5]. Its goal is to simplify the performing of various supervised machine learning experiments. In the following we shortly present the frameworks used in DKPro TC and describe their role in this framework.

Short description of the frameworks used in DKPro TC

Apache UIMA (*Unstructured Information Management Application*) is a software framework designed to analyze large volumes of unstructured information. It uses pipeline principle in order to perform its tasks. Each UIMA pipeline consists of several components. Input data, e.g. text, is represented as a CAS (*Common Analysis Structure*) object. This object is sent through all pipeline components. Each component takes the CAS object, extracts necessary information, writes the results back to CAS and makes it available to the next component. Classical UIMA data flow for text processing can be described in three steps:

1. *Reading input data*. Document is read from a collection and the corresponding CAS object is created. For this purpose the UIMA's CollectionReader is used.
2. *Processing the data*. Document is processed by different pipeline components, called *AnalysisEngines*. Each of these components usually annotate the part of the CAS with useful information.
3. *Writing processed data*. Processed and annotated document is saved in some data format for further processing/analysis, by using UIMA's Consumer component.

DKPro Core is a collection of state-of-the-art natural language processing components for Apache UIMA framework. Examples of such components are tokenizer, stemmer, lemmatizer, part-of-speech tagger, language identifier, spell corrector etc.

DKPro Lab framework is designed with the goal to perform parameter sweeping experiments. The experiments can be split into interdependent tasks. Each task has a set of parameters, which are injected using annotated class fields. The output data produced by each task for particular parameter configuration is stored and can be re-used to avoid the recalculation of results. The experiment results can be saved and presented by using the reporting functionality of the framework.

[1] https://code.google.com/p/dkpro-tc/

[2] http://uima.apache.org/

[3] https://code.google.com/p/dkpro-core-asl/ , https://code.google.com/p/dkpro-core-gpl/

[4] https://code.google.com/p/dkpro-lab/

[5] http://www.cs.waikato.ac.nz/~ml/weka/

Weka is a Java-based toolkit, which implements many state-of-the-art machine learning algorithms. It allows to quickly perform machine learning experiments on different data sets. It has easy-to-use API, flexible plugin mechanism as well as a graphic user interface. Weka includes algorithms for regression, classification, clustering, association rule mining and attribute selection. [HFH$^+$09].

Usage of frameworks in DKPro TC

DKPro TC framework incorporates the frameworks mentioned above and builds a powerful and easy-to-use platform for text classification experiments. The whole classification process is split into several DKPro Lab tasks, e.g. PreprocessTask, MetaInfoTask, ExtractFeaturesTask, TestTask etc. Each of these tasks is UIMA based, i.e. that the task is executed as a pipeline and the output of the one task is the input of the next task. The whole configuration of the framework is done by using DKPro Lab. DKPro Core components are mostly used for reading/writing between tasks as well as in different feature extractors (as annotations). Finally, Weka is currently used as the main machine learning classification platform, however the framework can be easily extended for other machine learning libraries.

In general, in order to create an experiment, a framework user just needs to implement a custom reader for her collection, a preprocessing pipeline and a configuration file. For preprocessing the pipeline it is convenient to use components from DKPro Core. In the configuration file a user usually only defines the classification algorithms she wants to try out and different combinations of feature sets. DKPro TC already comes with a lot of implemented features of general use, such as the number of sentence, n-grams, named entities ratio, part of speech ratio etc. However, if there is a need of more specific features, it can be easily implemented and added to the configuration file.

7.2 Performance measure

As a performance measure for our experiments we decide to use the well-known metrics from information retrieval: F1-measure, precision, recall and accuracy [Pow07]. The performance is evaluated for each class separately. For this purpose first we need to calculate true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN) classification instances for the particular class. These terms are defined as follows:

- TP - class is labeled as positive in the corpus and the classifier's prediction is also positive.
- TN - class is labeled as negative in the corpus and the classifier's prediction is also negative.
- FP - class is labeled as negative in the corpus but the classifier's prediction is positive.
- FN - class is labeled as positive in the corpus but the classifier's prediction is negative.

After determining each of these terms for a class, we can compute the metrics by using following formulas:

$$Precision = \frac{TP}{TP + FP} \tag{7.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{7.2}$$

$$F1\text{-}measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{7.3}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{7.4}$$

As we can see from these formulas, precision is the portion of correctly classified positive instances from all positive predicted instances. Recall is the fraction of correctly classified positive instances from all positive instances. F1-measure is a combination of precision and recall metrics and represents their harmonic mean. Accuracy shows the fraction of the correctly classified instances from all instances.

7.3 Classifiers used in the experiments

For our experiments we choose the following algorithms presented in the Weka toolkit: Naive Bayes, SMO, J48 and Random Forest. These algorithms are commonly used for text classification tasks [AZ12].

Naive Bayes (NB) is a simple classification algorithm, which is based on the theorem of probability known as Bayes rule. A characteristic feature of Naive Bayes algorithm is an assumption that all classification features are independent of each other. This is also referred to as the independent feature model. Despite the fact, that this independence assumption is clearly wrong for many real-world tasks, Naive Bayes often performs surprisingly well in various text classification tasks [MN98].

J48 is an open source implementation of the C4.5 algorithm. C4.5 is an algorithm which constructs decision trees from a set of training data. Starting with the root of the tree it splits the training data into subsets by using an attribute's value. The criterion used for splitting the tree is gain ratio and the attributes which are taken for splitting are the attributes with the highest gain [Qui96].

SMO (Sequential Minimal Optimization) is a support vector machine (SVM) learning algorithm. In the simplest linear form, an SVM tries to separate a set of positive examples from the negative ones by using a hyperplane. For the best classification performance, the separation should occur with a maximum margin. This margin, in the linear case, is defined as a distance from hyperplane to the nearest positive and negative examples. SVM can also perform a non-linear classification by implicitly mapping the input data to high-dimensional space using the so called kernel trick [Pla98].

Random Forest is an ensemble learning classification approach. The idea is to incorporate a group of weak classifiers in order to come up with a strong classifier. In this algorithm decision trees are used as weak classifiers. Random Forest constructs a number of decision trees, based on random subcollection of data and features, at training time and prediction is made by aggregation of the results produced by each decision tree. Because of the randomness and incorporation of ensemble learning the Random Forest classifiers do no overfit, have low variance and bias [Bre01].

7.4 Baseline and upper bound of experiments

In order to compare the results of our experiments we need to establish a baseline. Since, to the best of our knowledge, there are no publicly available results of similar experiments in german language, we decide to establish our baseline by taking a simple classifier and a simple set of features. Therefore, as a classifier for the baseline we use the Naive Bayes algorithm. As a feature set we just consider classical n-grams (unigrams, bigrams and trigrams). We do not apply any feature selection algorithms for establishing the baseline. The resulting baseline values are described in the corresponding experiments sections.

We also decide to establish a human performance. For this purpose we take one annotator as a gold standard and calculate the performance of two others, and then repeat this for each annotator. In order to get a single value for each label we compute the average value of each metric of three annotators. The results of these computations are shown in Table 7.1. Here we can see that annotators perform very good at labeling Arguments by Polarity and have trouble with labeling Arguments by Argumentative

Type (`Quantitative` label). However, the human performance in Arguments by Argumentative Type is lower than it could be because, as mentioned in Section 6.6.2, one annotator labeled only `Quantitative` arguments which are based on the results of studies and missed other ones.

Annotator	**Avg. measure value**	**Argumentative type**		**Reference**		**Polarity**	
		Quantitative	Qualitative	Referen.	Unreferen.	Pro	Contra
1	Precision	0.726	0.986	0.916	0.832	0.948	0.946
	Recall	0.937	0.927	0.829	0.917	0.946	0.947
	F1-measure	0.818	0.956	0.870	0.872	0.947	0.946
2	Precision	0.880	0.932	0.819	0.910	0.951	0.945
	Recall	0.731	0.974	0.911	0.818	0.946	0.951
	F1-measure	0.798	0.952	0.862	0.861	0.948	0.948
3	Precision	0.910	0.931	0.872	0.887	0.953	0.963
	Recall	0.739	0.980	0.881	0.880	0.963	0.953
	F1-measure	0.815	0.955	0.876	0.883	0.958	0.958
Average all	Precision	0.839	0.950	0.869	0.876	0.951	0.951
	Recall	0.802	0.960	0.874	0.872	0.952	0.950
	F1-measure	0.810	0.954	0.869	0.872	0.951	0.951

Table 7.1: Human performance of argument classification

We also compute the human performance on argument extraction task. As it is described in Section 7.6 we consider this task as a binary classification task of all sentences in each text. In such a way each sentence is either a part of an argument or not. By following same principle we compute the human baseline for the argument extraction task. The results are shown in Table 7.2.

Annotator	**Avg. measure**	**Argument**
1	Precision	0.590
	Recall	0.492
	F1-measure	0.536
2	Precision	0.572
	Recall	0.674
	F1-measure	0.619
3	Precision	0.635
	Recall	0.593
	F1-measure	0.613
Average all	Precision	0.599
	Recall	0.586
	F1-measure	0.589

Table 7.2: Human performance of argument extraction

We consider the human performance to be the upper bound for our experiments, since if the humans have problems in arguments classification/detection, it is likely that the automatic classifiers will face the same problems as well.

7.5 Experiments with argument classification types

In this section we describe three experiments with Arguments by Argumentative Type, Arguments by Reference and Arguments by Polarity. First we introduce the training and evaluation approach for this experiments. After that we describe each experiment in details.

7.5.1 Training and evaluation approach

In order to correctly evaluate the results we split our data in development and test set. Development set takes up 80% of all instances and the test set 20%. All our intermediate experiments such as, classifier tuning, feature tuning as well as feature selection are performed on the development set. For this purpose we decide to use 5-fold cross-validation. With 5 folds we get sufficient classification stability and fast execution at the same time.

After the tuning process, we choose the configurations with the best F1-measure value and carry out the end experiment by learning a model from the development set and using the test set for the evaluation. In such a way we avoid the overfitting which may occur after tuning the classifiers.

7.5.2 Experiments with Arguments by Argumentative Type

In this section we present the experiments on the classification of Arguments by Argumentative Type. First, we describe features used for the classification, then we tune our experiments in order to increase the performance and finally present the results.

Features

For this experiment we use a set of features presented in Table 7.3. First we use the top 500 uni-, bi- and trigrams that occur in all arguments. We consider these n-grams as binary features: we assign *true* if n-gram occurs and *false* if not. Furthermore, since `Quantitative` arguments tend to contain numbers, we take the amount of part of speech (POS) tags per all tokens in the argument as a feature. We are especially interested in the words with the tag CARD[6], which denotes cardinal numbers. For the extraction of POS tags we use a language independent POS tagger, called TreeTagger[7], which is already integrated in DKPro Core. The third kind of features is the number of named entities per argument sentence. We include these features because the arguments with the results of studies and polls frequently contain the name of the organization, which conduct these studies. For extraction of named entities we use the StanfordNER[8] component, which is also included in DKPro Core. The last feature, quantitative word list, is derived from the analysis of the words in `Quantitative` arguments. This list includes words such as: *"Umfrage"*, *"Studie"*, *"Votum"* etc. Each instance is checked for the occurrence of any word from the list. If an instance contains at least one word, the feature is assigned the value *true*, otherwise *false*.

Experiment tuning

After we get the first results of our experiments we perform further classification analysis. We notice that the data used in these experiments is imbalanced (`Quantitative` 77.2% to `Qualitative` 22.8 % Section 6.7). This problem may cause negative effects on classification performance [Gan12]. The possible

[6] `https://code.google.com/p/dkpro-core-asl/wiki/UniversalPosMapping1_5`

[7] http://www.cis.uni-muenchen.de/ schmid/tools/TreeTagger/

[8] http://nlp.stanford.edu/ner/

Feature	Description
N-grams	Top 500 of unigrams, bigrams and trigrams
POS ratio	Amount of different POS tags per all tokens in the argument.
Named entities ratio	Amount of different named entities (Organization, Person, Location) per argument sentence.
Quantitative word list	A list of words that characterize the `Quantitative` arguments, e.g. *"Umfrage"*, *"Studie"*, *"Votum"* etc.

Table 7.3: Features used for classification of Arguments by Argumentative Type

solutions to this problem can be performed on the data or algorithmic level. One of the most popular data level solutions is so-called resampling: either by over-sampling the minority class or by under-sampling the majority class. However, the first one may introduce overfitting and the second does not work well when the minority class has not enough instances (which is our case). Therefore, we decide to use a more sophisticated approach on the algorithmic level by using the Cost-Sensitive learning. This type of learning takes the misclassification costs into consideration and is used as a meta classifier (in Weka CostSensitiveClassifier) over the actual classifier. For this purpose we determine the cost ratio by inverting the class distributions and rounding them to the nearest integer (1:4) [TNGST10]. By applying this technique we significantly increase the performance of the SMO classifier by 11%.

Furthermore, we also tune the parameters of classification algorithms. By changing the confidence factor used for pruning in J48 we obtain a small 1% increase in performance. By changing the kernel in SMO from polynomial to RBF [HCL03] we improve the performance by 6%.

We also apply two feature selection techniques, the χ^2 metric and the Information Gain [For03]. In our experiments we try out the top 100, 200 and 300 features by using each of this metrics. We obtain the best values by using the top 100 features. For the Information Gain metric we obtain slightly better results than with χ^2. By using feature selection we increase our performance by 1-2%, depending on the classifier.

Results

Table 7.4 presents the best results of our experiments evaluated on the 5-fold cross-validation development set (Dev.) and on the test set (Test). We achieve the best performance by combining the meta Cost Sensitive classifier with the SMO. However, without the cost correction the SMO classifier showed the worst results, i.e. SMO is very sensitive to imbalanced data. Naive Bayes shows the same results with and without the meta classifier and J48 together with Random Forest perform even worse on the test set when used with the meta classifier.

We also compare our results to the baseline and the human performance. We significantly outperform our baseline with the best classifier by 17% for `Quantitative` arguments and by 7% for `Qualitative`. Moreover, all the classifiers outperform the baseline on the test set. It gives us confidence in the features we use as well as the tuning we perform. We are also able to outperform the upper bound of the classification - human performance, in average by 6.9%. However, as we mentioned in Section 7.4, the human performance for these experiments could be slightly lower than it actually is, because of the misclassification caused by one of the annotators. Therefore, we decide to compute the human performance by using only two annotators and we obtain the following values: `Quantitative` 0.835 F1-measure and `Qualitative` F1-measure 0.958. This slightly changes the picture, but we still outperform the human by 5.4% in average.

We also observe that the results on the development set are significantly worse than on the test set (e.g. for SMO up to 35%). This is, however, not expected behavior, since we tune our classifiers and features on the development set which may lead to overfitting and the test set remains unknown to the classifier

Classifier	Tuning	Quantitative		Qualitative	
		Dev.	Test	Dev.	Test
Baseline (NB + ngrams)	-	0.632	0.750	0.869	0.906
Human	-	0.810	0.810	0.954	0.954
NB	InfoGain 100	0.738	0.826	0.920	0.939
NB + Cost Sensitive	InfoGain 100	0.738	0.826	0.915	0.939
J48	InfoGain 100 / C=0.1	0.774	0.878	0.941	0.963
J48 + Cost Sensitive	InfoGain 100 / C=0.1	0.776	0.818	0.926	0.939
RandomForest	InfoGain 100	0.714	0.857	0.933	0.955
RandomForest + Cost Sensitive	InfoGain 100	0.778	0.829	0.938	0.948
SMO	InfoGain 100 / RBF Kernel	0.439	0.788	0.905	0.951
SMO + Cost Sensitive	**InfoGain 100 / RBF Kernel**	**0.826**	**0.923**	**0.946**	**0.978**

Table 7.4: F1-measure performance of the classification of Arguments by Argumentative Type

during the tuning. One of the possible reasons for that is the lack of data in the test set, especially if the test set is imbalanced (only 20 Quantitative arguments). This can cause a poorly chosen split to provide exceptionally good (or bad) results. In order to prove the feasibility of this statement we decide to conduct another experiment by taking all available data and performing the 10-fold cross-validation. If our statement is wrong then the performance of the 10-fold cross-validation should not be significantly worse than the performance on the test set. However, the experiment, presented in Table 7.5, shows the difference in performance for Quantitative arguments by 11%. We consider this result to be a confirmation of our statement about the poorly chosen split.

Despite the probably inadequate chosen split for this experiment, according to the results presented in Table 7.5 we are able to reach the human performance, with the 10-fold cross-validation experiment, for Quantitative label and lose only 1% for Qualitative label.

Classifier	Tuning	Quantitative	Qualitative
Baseline (NB + ngrams)	-	0.701	0.902
Human	-	0.810	0.954
NB	InfoGain 100	0.741	0.920
NB + Cost Sensitive	InfoGain 100	0.752	0.919
J48	InfoGain 100 / C 0.1	0.762	0.933
J48 + Cost Sensitive	InfoGain 100 / C 0.1	0.714	0.908
RandomForest	InfoGain 100	0.720	0.931
RandomForest + Cost Sensitive	InfoGain 100	0.772	0.938
SMO	InfoGain 100 / RBF Kernel	0.771	0.942
SMO + Cost Sensitive	**InfoGain 100 / RBF Kernel**	**0.813**	**0.944**

Table 7.5: F1-measure performance of the classification of Arguments by Argumentative Type (10-fold cross-validatation)

Among all features used in the experiments, the most predictive, according to the Information Gain, are the ratio of POS tag *CARD*, occurrence of words from the quantitative word list and the unigrams, such as *"prozent"*, *"euro"*, *"befragten"* etc. Surprisingly, the named entity features do not contribute to the classification at all. We analyze this aberration and assume that the reason for this could be the low performance of StanfordNER on the given dataset for german language. Here are the few examples, when it fails: *"...liegt nun mit der "Kess 12-Studie"..."*, *"49 Prozent aller Lehrer sagen laut Allendsbach-Studie..."*. The words as *"Kess 12-Studie"* and *Allendsbach-Studie* are not recognized as named entities.

Error analysis

After undertaking the various experiments we also perform an error analysis by taking a look at the misclassified instances. According to our analysis, the most frequent error source for the `Quantitative` label is the presence of different numbers in the argument, which are not important for argumentation. Consider the following examples: *"Ich habe im* **achten** *Schuljahr eine freiwillige Ehrenrunde gedreht..."*, *"Ihnen fallen vor allem* **zwei** *besondere Lebenssituationen..."* , *"Motiv für jeden Promotionswilligen auch die Aussicht auf* **zwei** *Buchstaben mehr im Reisepass..."*. All these parts of the argument contain numbers which are irrelevant for argumentation, but since one of the most predictive features is the occurrence of numbers in the argument, this possibly causes a misclassification behavior.

We do not find a particular error pattern for `Qualitative` labels, since there is only a small number of instances which are misclassified (because of the high classification performance). Therefore, we just present two examples of individual cases:

- Synonyms for numbers (50% - a half), e.g. *"Mehr als die Hälfte der Bürger..."*. "Hälfte" is not recognized as a number.

- Arguments with long texts and only one number in them. This may introduce noise due to n-grams and the occurrence of one number is not enough to outperform the noise of n-grams.

7.5.3 Experiments with Arguments by Reference

In this section we present the experiments on the classification of Arguments by Reference. First we describe features used for the classification, then we tune our experiments in order to increase the performance and finally present the results.

Features

For this experiment we use a set of features presented in Table 7.6. First we use the top 500 uni-, bi- and trigrams that occur in all arguments as binary features (either *true* if occurs, *false* if not). Additionally, since `Referenced` arguments should contain a particular source, which is usually a person or organization we decide to include named entities as a feature. We also think that the punctuation marks would be nice to have as a feature: many referenced arguments have different quotation marks, comas or colons. Therefore, we include POS tags as features (POS tag PUNC[9] has the information we actually need). The last feature is just a specific case of punctuation marks: quotation marks. We choose the quotation marks as a separate feature, since it is a characteristic property of the direct citations.

Feature	**Description**
N-grams	Top 500 of unigrams, bigrams and trigrams
Named entities ratio	Amount of different named entities (Organization, Person, Location) per argument sentence.
POS ratio	Amount of different POS tags per all tokens in the argument.
Quotation marks ratio	Amount of quotation marks per argument sentence

Table 7.6: Features used for classification of Arguments by Reference

[9] https://code.google.com/p/dkpro-core-asl/wiki/UniversalPosMapping1_5

Experiment tuning

After conducting the first experiments we perform an error analysis. First, we look into the classified data in order to find a possible source of misclassification. We notice that a lot of instances, which source is outside of the argument are misclassified. The reason for this is that we extract our features only from the arguments and do not consider the argument context. However, the framework which is used for conducting the experiments does not support the inclusion of context while working with text instances. Therefore, first we modify the DKPro TC framework and implement the functionality we require. After that we define two new features: distance to quotation mark and context n-grams (Table 7.7). The first feature is an ordinal feature, which is defined as the distance in sentences from the argument boundary to the first occurrence of a quotation mark in argument context. With this feature we are able to increase the performance by 2-4% depending on the classifier. The second feature are the n-grams which occur in the context of an argument. As a context, first, we define the next and previous three sentences from the argument. However, we achieve the best results by reducing the context to two sentences. We are able to improve the performance by 1-2% depending on the classifier.

Feature	**Description**
Distance to quotation mark	Distance, in sentences, from the argument boundary to the first quotation mark outside the argument (in argument context)
Context n-grams	Unigrams, bigrams and trigrams extracted from the next and previous two sentences

Table 7.7: List of additional features used for classification of Arguments by Reference

Furthermore, we experiment with the amount of top n-grams extracted for the n-gram feature vector. Initially we use 500 n-grams, however by increasing this amount to 1000 and 1500 (depending on classifier) we also increase the classification performance. It seems that infrequent words are able to contribute to the prediction.

We also experiment with the different classifier parameters. We increase the performance of the SMO algorithm by using the RBF kernel and complexity parameter `C` of 16. These changes improve the classification performance of SMO by 9%.

Results

Table 7.8 presents the best results of our experiments evaluated on the 5-fold cross-validation development set (Dev.) and on the test set (Test). We achieve the best performance on the test set by using Naive Bayes classifiers. However, Random Forest performs almost as well as Naive Bayes, the difference reaching only 1%. SMO, in its turn, lose 1% to Random Forest. The worst classifier according to the F1-measure on the test set is the J48.

All our classifiers are able to outperform the baseline. With Naive Bayes the difference reaches about 7.2% in average for both test and development sets. We do not outperform the upper bound of the classification, but we are very close to it, loosing only 1.8% in average for both labels.

We also observe, like in the classification of Arguments by Argumentative Type, that the performance on the test set is better than on the development set. However, the difference is not as significant as in the previous experiment, reaching in average 4%. Nevertheless, we still conduct the same 10-fold cross-validation experiment including all the data. Table 7.9 shows the results of this experiment. Here, we see a tendency that the performance values are approximately the same as in the development set. This indicates, that the values we obtain on the test set might have a little overestimation of the actual picture. However, all the classifiers are still able to outperform our baseline.

Classifier	Tuning	Referenced		Unreferenced	
		Dev.	Test	Dev.	Test
Baseline (NB + ngrams)	-	0.734	0.766	0.712	0.796
Human	-	0.869	0.869	0.872	0.872
NB	**InfoGain 300, 1500 n-grams**	**0.783**	**0.843**	**0.809**	**0.863**
J48	InfoGain 300, 1000 n-grams, C 0.1	0.776	0.787	0.812	0.787
RandomForest	InfoGain 300, 1500 n-grams	0.803	0.833	0.808	0.851
SMO	InfoGain 300, 1000 n-grams, RBF Kernel with C 16	0.754	0.815	0.805	0.845

Table 7.8: F1-measure performance of the classification of Arguments by Reference

As we expected, among the best ranked features, is the number of person/organization entities, quotation mark ratio as well as punctuation mark ratio. Additionally, distance to quotation mark has proven to be useful. The most predictive n-grams are the quotation marks, words as *"sagen"*, *"meinen"*, *"ich"*, *"wir"*. This is not surprising, since these words either tend to be used before a citation (*"sagen"*, *"meinen"*) or directly in the citation (*"ich"*, *"wir"*). Context n-grams look the same as the argument n-grams, however, they do not contribute a lot to the classification (Information Gain 1-2%). The reason for this might be noise that they can introduce. For example, if a classified argument is not referenced, but the sentences before or after this argument are, it can lead to misclassification.

Classifier	Tuning	Referenced	Unreferenced
Baseline (NB + ngrams)	-	0.747	0.751
Human	-	0.869	0.872
NB	**InfoGain 300, 1500 n-grams**	**0.776**	**0.816**
J48	InfoGain 300, 1000 n-grams	0.791	0.799
RandomForest	InfoGain 300, 1500 n-grams	0.790	0.794
SMO	InfoGain 300, 1000 n-grams, RBF Kernel with C 16	0.764	0.804

Table 7.9: F1-measure performance of the classification of Arguments by Reference (10-fold cross-validatation)

Error analysis

We perform an error analysis for the misclassified instances by looking at the classification results. According to our analysis, the most frequent error source for the experiments with Argument by Reference is the inaccuracy in detecting the named entities. The reason for this is the low performance on our dataset of StanfordNER, which is used for detecting named entities. This also confirms our assumption in Section7.5 (Results), where we observe low prediction efficiency of the named entity feature. Here are some examples where StanfordNER failed to recognize entities: *"Laut OECD liegen... "*, *"Die TU9-Allianz, ein Verband... "*, *"Saurin bezweifelt, dass diesen Kindern geholfen ist..."*. In the following sentence the word "Euro" is recognized as an organization entity, but in this case it refers to a currency: "Jahr fast eine Milliarde Euro". These prediction errors of named entities are crucial for the classification of Arguments by Reference, since the most predictive feature for this kind of experiments is the number of person/organization entities (describe in previous section). By improving the performance of the named entity recognizer we could improve the classification accuracy in our experiments.

Another source of prediction errors is concerned with the feature quotation mark ratio. This is one of the characteristic features of `Referenced` arguments. However, some of the `Unreferenced` arguments may contain it as well, which leads to classification problems. The examples of such cases are: *"kann ein „Dr." der Karriere dienen"* , *"Im Förderschwerpunkt „Sprache" wissen die Eltern den therapeutischen Anspruch der „Durchgangsschule" "*, etc. Here we just have single words separated by quotation marks, which is obviously not a citation. These errors could be avoided by using either regular expression, which replaces quotation marks for short phrases with some special character or by introducing a new feature: text length between quotation marks.

7.5.4 Experiments with Arguments by Polarity

In this section we present experiments on the classification of Arguments by Polarity. First, we describe features used for the classification, then we tune our experiments in order to increase the performance and finally present the results.

Features

For this experiment we use a set of features presented in Table 7.10. As in previous experiments, we take the top 500 uni-, bi and trigrams that occur in all arguments as binary features (either *true* if such an n-gram occurs, *false* if not). Furthermore, we analyze the characteristics of arguments in our dataset and come to the conclusion that classification of Arguments by Polarity can be reduced to polarity classification concerning the argument target. Therefore, we also analyze similar works regarding polarity classification on sentence level [JP13] [WK09] [WWH05]. As a result of this analysis we choose additional features for our experiment, such as: number of positive/negative words, number of positive/negative adjectives, number of positive/negative nouns, number of positive/negative verbs and number of positive/negative words on different depth level of the syntax tree. The last feature requires sentence preprocessing. For this purpose we use Berkeley Parser[10], which is already provided in DKPro Core. Additionally to the polarity features mentioned above we want to handle negations, since it completely changes the polarity of a sentence. For this purpose we introduce a list of polarity changer words (e.g. *"nicht"*, *"kein"* etc.). We consider this feature as a binary feature: it is *true* when an argument contains one of these words and *false* if not.

In order to determine which polarity has a particular word we need to use a resource for sentiment analysis. GermanPolarityClues[11] and SentiWS[12] are one of the best resources for this purpose when using German language [Mom12]. We obtain slightly better results by using the GermanPolarityClues, the reason for that might be a broader coverage of the words: more than 10000 against around 3500.

Experiment tuning

In our first experiments we do not specifically consider a target for each argument, but take a simple assumption that the polarity target of all arguments is the document title. This, however, is not always the case. Consider the following example: *"Promovierte der Uni Köln verdienen ein bis zwei Jahre nach ihrem Abschluss durchschnittlich 21,21 Euro brutto pro Stunde, während Masterabsolventen nur 17,90 Euro*

[10] https://code.google.com/p/berkeleyparser/

[11] `http://www.ulliwaltinger.de/sentiment/`

[12] `http://asv.informatik.uni-leipzig.de/download/sentiws.html`

Feature	Description
N-grams	Top 500 of unigrams, bigrams and trigrams
Adverbs	Argument adverbs as binary feature (*true/false*)
Verbs	Argument verbs as binary feature (*true/false*)
Occurrence of polarity changers	A list of polarity changers (e.g. *"nicht"*, *"kein"* etc.). Used as a binary feature (*true/false*)
Number of positive/negative words	Number of positive and negative words in an argument
Number of positive/negative adjectives	Number of postive and negative adjectives in an argument
Number of positive/negative nouns	Number of postive and negative nouns in an argument
Number of positive/negative verbs	Number of postive and negative verbs in an argument
Number of positive/negative words on different depth level	Number of positive and negative words on different depth level of syntax tree

Table 7.10: Features used for classification of Arguments by Polarity

verdienen."[13]. This has positive meaning for the target *"Promovierte"*, but negative for *"Masterabsolventen"*. This is also crucial for documents with topics *"G8"* and *"G9"*, since the documents used for these topics are the same, but the target is different. However, we decide to exclude these topics from the classification of Arguments by Polarity, since targets *"G8"* and *"G9"* rarely occur in texts, usually synonyms are used. However, the synonyms of these terms are really difficult to resolve, e.g. *"Abitur nach 12 Jahren"* (*"G8"*), *"von früher"* (*"G9"*), *"verkürzte Gymnasialzeit"* (*"G8"*). Even for humans without some prior knowledge is not clear that *"Abitur nach 12 Jahren"* is actually a synonym for *"G8"* and *"von früher"* means *"G9"*.

Feature	Description
Polarity of words close to the argument target	Difference between positive and negative words that occur within a window of three words close to the argument target. If the argument target is missing the value of this feature is 0

Table 7.11: Additional feature used for classification of Arguments by Polarity

Therefore, as an additional feature for our experiments we decide to use the polarity of words that are close to the argument target (Table 7.11), if the target occurs in the argument. Polarity is here referred to as the difference between positive and negative words within the 3 word window. If the target is missing in the argument the value of this feature is 0. However, as we observe, the argument target frequently has a lot of different synonyms. For example in documents on the topic *"Sitzenbleiben abschaffen"*, we find following synonyms for the word *"Sitzenbleiben"*: *"Klassenwiederholung"*, *"Ehrenrunde"*, *"Wiederholerjahr"* etc. In order to increase the probability of finding the target in an argument these synonyms should be determined automatically. As a simple solution for this problem we decide to use a lexical-semantic resource, called GermaNet[14]. It can provide us with a list of synonyms and antonyms for a given word. However, this resource is of general use, and does not contain a lot of synonyms for educational topics (e.g. *"Klassenwiederholung"*, *"Ehrenrunde"*, *"Wiederholerjahr"* is missing there as synonyms). Therefore, as an additional source of target synonyms we decide to use the Tf-Idf metric for nouns. This metric is frequently used in keyphrase extraction algorithms, since the words with

[13] http://www.ksta.de/job-und-karriere/promotion-mehr-geld-mit-doktortitel,20063080,22559908.html
[14] http://www.sfs.uni-tuebingen.de/lsd/

the greatest Tf-Idf value are the most characteristic words for a particular document [PNPH08] [EGR13]. We incorporate this principle by calculating the Tf-Idf value only for nouns in arguments, assuming that the most characteristic nouns in an argument are the argument target synonyms. For this purpose we modify our dataset as follows: for each document we extract text parts corresponding to arguments and create a new pseudo document out of them. By using this technique we are able to recognize more synonyms for an argument target. By using the polarity of words close to the argument target feature we are able to increase the performance by 5-8% depending on the classifier.

We also perform an error analysis on the development set. This yields us new ways of improvement. We observe, that the polarity of some words we use in the sentiment corpus differs from the real polarity in our dataset. For example such words as *"nötig"*, *"notwendigkeit"*, *"brauchen"* are negative according to the sentiment corpus, but it is positive for our dataset e.g. *"... man braucht den Master, um die Karrierechancen zu verbessern... "*. Therefore, we decide to adapt the sentiment corpus for our dataset, by changing the polarity of several words. This changes further imporove the classification performance by 2-3% depending on the classifier.

Furthermore, we experiment with the amount of top n-grams extracted for the n-gram feature vector, and apply Information Gain and χ^2 feature selection techniques. We also tune the classifier parameters. For the SMO classifier we increase the performance by using the RBF kernel and complexity parameter C of 16. We get an improvement of 5%.

Results

Table 7.12 shows the best results of our experiments evaluated on the 5-fold cross-validation development set (Dev.) and on the test set (Test). We reach the best performance on the test set by using the SMO classifier. Naive Bayes and Random Forest show slightly lower results and J48 performs worse then the other. We also conduct an experiment with the best configuration for SMO classifier on a dataset, where G8/G9 documents are included. As we expect the inclusion of these topics introduces additional noise to the classification, and we obtain results even worse than with J48 classifier on the "clean" set.

Classifier	**Tuning**	**Pro**		**Contra**	
		Dev.	Test	Dev.	Test
Baseline (NB + ngrams)	-	0.619	0.659	0.580	0.559
Human	-	0.951	0.951	0.951	0.951
NB	χ^2 200, no n-grams	0.707	0.684	0.733	0.675
J48	χ^2 200, no n-grams	0.645	0.649	0.696	0.658
RandomForest	InfoGain 100, 1000 n-grams	0.679	0.667	0.715	0.691
SMO (with G8/G9)	InfoGain 100, 2000 n-grams, RBFKernel with C 16	0.600	0.612	0.672	0.692
SMO	**InfoGain 100, 2000 n-grams, RBFKernel with C 16**	**0.676**	**0.686**	**0.712**	**0.729**

Table 7.12: F1-measure performance of the classification of Arguments by Polarity

All our classifiers are able to outperform the baseline. By using SMO the difference reaches about 9% in average for both test and development sets. However, we do not outperform the upper bound of the classification. The human performance is really high in identifying pro/contra argumentation. The possible reasons for such a significant difference between human and automatic performance are described in the following section.

Unlike in previous experiments the difference between the performance on the test set and development set are not really significant. However, for consistency we also conduct the 10-fold cross-validation experiment including all the data. Table 7.13 depicts the results of this experiment. These results correlate with the results from our initial experiment for the test and development set. This indicates, that the results obtained from the experiment on the test set are depicting a real picture.

Classifier	**Tuning**	**Pro**	**Contra**
Baseline (NB + ngrams)	-	0.633	0.592
Human	-	0.951	0.951
NB	χ^2 200, no n-grams	0.667	0.706
J48	χ^2 200, no n-grams	0.626	0.717
RandomForest	InfoGain 100, 1000 n-grams	0.656	0.716
SMO (with G8/G9)	InfoGain 100, 2000 n-grams, RBFKernel with C 16	0.589	0.693
SMO	**InfoGain 100, 2000 n-grams, RBFKernel with C 16**	**0.670**	**0.721**

Table 7.13: F1-measure performance of the classification of Arguments by Polarity (10-fold cross-validatation)

The most predictive features are the number of positive/negative words, occurrence of polarity changers as well as polarity of words close to the argument target. Additionally, different adverbs and verbs of the argument do a valuable contribution. Number of positive/negative adjectives/nouns/verbs and word on different depth level provide high level of Information gain, however have no influence on the overall performance. This indicates, that these features are conditionally dependent from other features we have in the feature set.

Error analysis

The most frequent source of misclassification is the fact, that we consider all words in isolation. We calculate the amount of positive and negative words and check for occurrence of negations. However, with this approach we loose important structure information. Consider the following example: *"In manchen Ländern ist Doktorandenstudium als lohnend, spannend und nicht sehr zeitaufwendig angesehen"*. We have two positive words ("lohnend", "spannend") and one negative ("zeitaufwendig"), this means that the general polarity is positive, but the occurrence of negation changes it to negative (according to our approach). This, however, can be avoided by using the meaning of longer phrases instead of single words. One of the state of the art works dealing with this problem for English language is [SPW$^+$13]. It was recently published on the EMNLP[15] conference. This approach could be also adapted for the German language, and may improve the performance of experiments with Arguments by Polarity.

Another source of misclassification is the occurrence of idioms in the arguments, e.g. *"könne er die Raten für sein Haus nicht stemmen"*, *"zunehmend an den Rand gedrängt"*, *"die Unternehmen schmücken sich gerne mit"* etc. In order to correctly resolve the polarity of such phrases, a special sentiment resource needs to be created.

The last error pattern is the peculiarity of the German language to have verbs with separable prefixes (e.g. "zunehmen" ->"nimmt zu"). Consider the following sentence: *"Eine Leistungsverbesserung bleibe bei den meisten Klassenwiederholern allerdings aus."*. Here we have one verb with separable prefix "ausbleiben" and it has negative polarity, however in the separate form we get two separate words ("bleiben" and "aus"), which meaning in general is not negative since we are looking at these words in isolation.

[15] http://hum.csse.unimelb.edu.au/emnlp2013/

7.6 Argument extraction experiment

In this section we present the argument extraction experiments . First, we describe our approach for this extraction task. After that we introduce training and evaluating methodology, introduce the features and describe the experiment tuning. Finally, we present the results of the experiments and perform an error analysis.

7.6.1 Extraction as classification task

The extraction of arguments presented in the document is considered to be a binary classification task. Each sentence is classified to be a part of an argument or not. In this way all sentences classified as a part of an argument form together all the arguments of the document. This is an idea adapted from the following paper [PM09]. However, this method presents a segmentation problem, since the delimiters of each argument are not defined. For our dataset this is not, however, a very critical issue, since the majority of arguments are separated from each other by non-argumentative parts of text. Therefore, to solve this problem we apply a simple approach and consider following argument sentences to form one whole argument.

7.6.2 Training and evaluation approach

From our dataset we obtain 824 argumentative sentences. We extract another 941 sentences at random from the text, which do not belongs to arguments, and use them as negative examples. Therefore, for these experiments we use 1648 classification instances. In order to avoid overfitting and obtain reliable results we split this data in a development and test set. The development set contains 80% of all instances and the test set 20%. On the development set we perform our intermediate experiments such as, classifier tuning, feature tuning and feature selection. As a classification method for intermediate experiments we choose 20-fold cross-validation. After this we choose the configurations with the best F1-measure value and execute the final experiment. For this experiment we learn a model on the development set and use the test set for the evaluation.

7.6.3 Features

For the argument extraction experiment we use a set of features presented in Table 7.14. All these features are taken from a similar work of Moens and Palau et al. [MBPR07]. First we use the top 500 uni-, bi- and trigrams as binary features (*true,false*). The adverbs and verbs detected by a part-of-speech tagger are used as binary features as well (*true/false*), since they can signal argumentative information [MBPR07]. Additionally, we include modal verbs as a binary feature, which indicates if a modal verb is present. Modal verbs show the level of necessity. We also consider different text statistic features such as sentence length - arguments may consist of more words than regular sentences; average word length - longer words may occur in argumentative sentences; number of punctuation marks - the presence of argumentation might increase the number of punctation marks. The last feature is the presence of words indicative for argumentation. The examples of such words are *"deshalb"*, *"aber"*, *"folgen"* etc. This is used as a binary feature. Moens and Palau et al. obtain this list from another work [KD93] and for English language. We, however, use German language, therefore we decide to create a similar list on our own. First, we translate some parts of the most important words from this list into German. After that, we analyze the arguments from our dataset and select additional words. The selection is performed by two annotators and we consider only the words on which both annotators agreed.

Feature	Description
N-grams	Top 500 of unigrams, bigrams and trigrams
Adverbs	Argument adverbs as binary feature (*true/false*)
Verbs	Argument verbs as binary feature (*true/false*)
Modal verbs	List of German modal verbs as binary feature (*true/false*)
Sentence statistics	It includes average word length, sentence length, number of punctuation marks
Words indicative for argumentation	List of 100 words which are indicative for argumentation. Examples: *"deshalb"*, *"aber"*, *"folgen"* etc.

Table 7.14: Features used for argument extraction

7.6.4 Experiment tuning

After conducting first experiments we analyze the possible points of improvement for the classification. As we observe in Section 7.5.4 the majority of arguments in our dataset contain a lot of sentiment words. Therefore, we decide to reuse this peculiarity in this experiment by taking a number of positive/negative words in a sentence as a feature. The usage of these features improves the performance in average by 1-3% depending on the classifier. As an additional point of improvement we decide to take context into account. The reason for this is that the adjacent sentences are usually logically connected with each other in a free text. As context features we take the n-grams of the preceding and following sentences as well as the occurrence of words indicative for argumentation. However, these context features do not provide us with any performance improvement. The above described features are summarized in Table 7.15.

We also experiment with other feature and classifier parameters. By increasing the amount of top n-grams used in the classification to 1500 and by using the χ^2 feature selection method with 200 features, we are able to increase overall performance of classifiers. However, tuning the classifier parameters does not provide us with significant improvement, only for the SMO classifier we increase the performance by using the RBF kernel.

Feature	Description
Number of negative/positive words	Number of positive and negative words in a sentence
N-grams in context	unigrams, bigrams, trigrams in sentence context (preceding and following sentence)
Words indicative for argumentation in context	Occurrence of words indicative for argumentation in previous and following sentence

Table 7.15: Additional features used for argument extraction

7.6.5 Results

Table 7.16 presents the best results of our experiments evaluated on the 20-fold cross-validation development set (Dev.) and on the test set (Test). We reach the best performance on the test and development set by using the Naive Bayes classifier. Other classifiers perform surprisingly bad, not even reaching the baseline on the test set. However, performance of the SMO on the development set is 3.6% greater than on the test set. It may indicate overfitting which occurred on the development set during the tuning.

We are also able to outperform the human performance. However, even the baseline we established performs better than the human one. This might be an evidence of a lower agreement of humans for this task.

Furthermore, we compare our performance to the work of Moens et al. [MBPR07]. In this work the authors provide only the accuracy of their experiments, therefore we calculate the accuracy for our experiments as well (Table 7.16). As we can see their results outperform our by 7.4%. However, what must be taken into account is that they already achieve an accuracy of 73.06% by using simple n-grams and all their features are able to improve this result only by 0.69%. We outperform the n-gram baseline by 3.4%, which indicates better quality of our features and the tuning we conduct in comparison to [MBPR07]. The reason for difference in the baseline performance could be the corpus and differences in English and German languages. The Moens et al. use the well-known Araucaria corpus with highly structured data. This corpus is already used in several experiments and exists since 2004. Our corpus is new, does not contain argument structure information and is used for experiments for the first time.

Classifier	**Tuning**	**Argument**		
		Dev.	Test	Accuracy
Baseline (NB + ngrams)	-	0.607	0.637	0.630
Human	-	0.589	0.589	-
NB	**1500 n-gram, χ^2 200**	**0.673**	**0.673**	**0.664 (+0.034)**
J48	1000 n-gram, χ^2 200	0.603	0.581	0.615
RandomForest	1500 n-gram, χ^2 200	0.613	0.626	0.612
SMO	1500 n-gram, χ^2 200, RBFKernel	0.640	0.604	0.603
[MBPR07]	-	-	-	0.738 (+0.0069)

Table 7.16: F1-measure performance of the argument extraction experiment

Among all features used in the experiments, the most predictive, according to the χ^2 metric are the number of negative/positive words, presence of words indicative for argumentation, occurrence of modal verbs as well as unigrams. Additionally sentence length has proven to be useful. Adverbs, verbs and context features turn out to have little influence on the prediction.

7.6.6 Error analysis

After undertaking the experiments we also perform an error analysis by taking a look at the misclassified instances. The most common error pattern we observe concerns sentences, which one annotator classifies as being argumentative during the annotation process while the other two annotators disagree. However, some of these sentences could be considered as premises for an argument. The classifier catch these cases, however, since this data was not labeled in the corpus it leads to the decrease of the performance. This problem could be avoided by excluding such sentences from the set of negative examples.

8 Front-end interface

In this chapter we introduce and describe a prototype of a user interface (UI) of the proposed search engine, referred to as MyPoint.

Figure 8.1 shows the UI of the developed prototype. We have here a classical text input field and a search button. The input field is used for entering the topic a user is looking for. On the left side of the UI we can see a list of hot topics. A user can select one of these topics and it will be automatically entered into the input field and searched for the results. This list has two main purposes. First is to provide a user with the first impression of what can be entered in the search field. Since when the page is opened for the first time it is not clear what kind of information is expected in the input field. The hot topic list provides this indirect instruction to the user. The second purpose of this list is to depict what topics are currently intensively discussed in the Web, which, we find, might be interesting for a possible user of the system. We propose to construct this list by calculating the amount of arguments for a particular topic for the period of one month. In this way the user is provided with highly controversial topics for the current month.

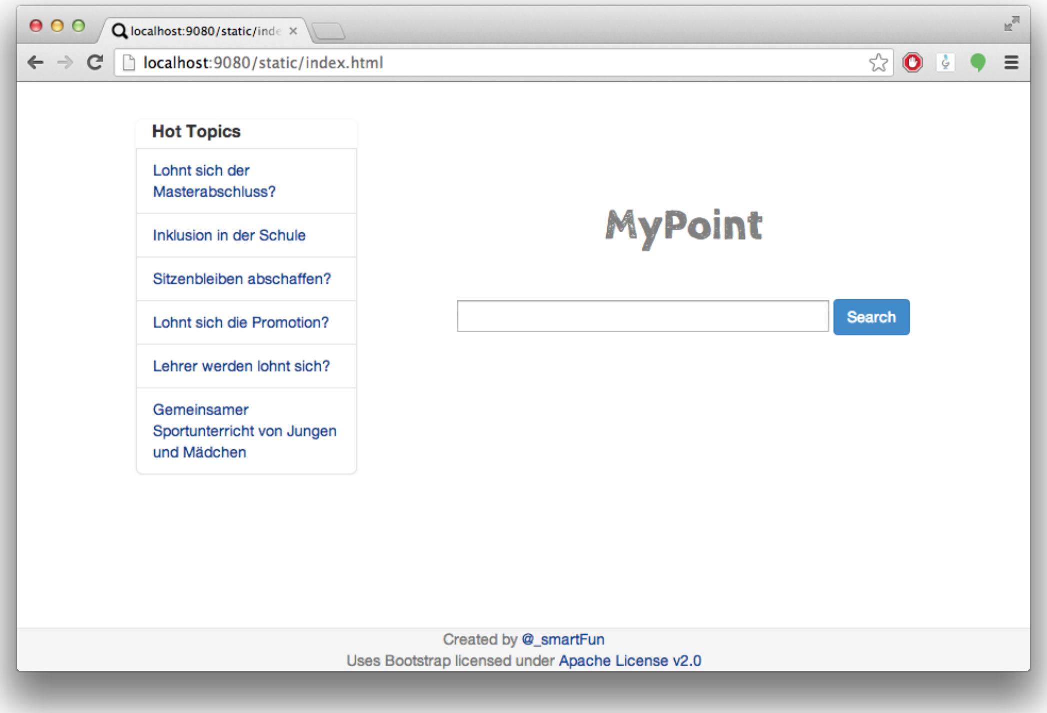
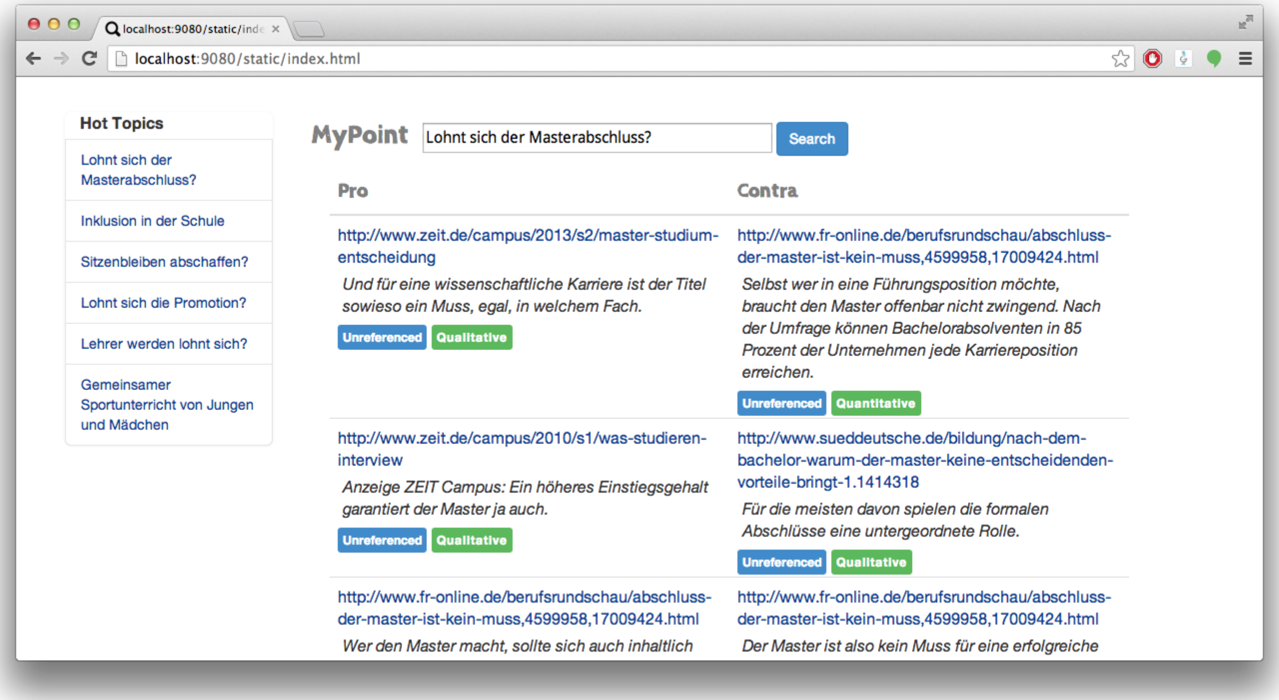

localhost:9080/static/inde
localhost:9080/static/index.html
Hot Topics
Lohnt sich der Masterabschluss?
Inklusion in der Schule
Sitzenbleiben abschaffen?
Lohnt sich die Promotion?
Lehrer werden lohnt sich?
Gemeinsamer Sportunterricht von Jungen und Mädchen
MyPoint
Search
Created by @_smartFun
Uses Bootstrap licensed under Apache License v2.0

Figure 8.1: Main user interface of MyPoint

Figure 8.2 demonstrates the UI of MyPoint after a user entered the topic she is looking for. The input field as well as the search button are animated to the top of the window and in the middle we can see the search results. They are split into two columns: `Pro` arguments and `Contra` arguments. This provides a user with a comparison style view, where she can read and analyze supportive and opposing argumentation at once. Last but not least, each entry in these columns contains an argument text, an URL to the original page, where the argument is extracted as well as two labels which describe the current argument either to be `Quantitative/Qualitative` or `Referenced/Unreferenced`. In this way we provide a user with all necessary information, which should help her by the decision making.



localhost:9080/static/index.html

Hot Topics

- Lohnt sich der Masterabschluss?
- Inklusion in der Schule
- Sitzenbleiben abschaffen?
- Lohnt sich die Promotion?
- Lehrer werden lohnt sich?
- Gemeinsamer Sportunterricht von Jungen und Mädchen

MyPoint Lohnt sich der Masterabschluss?

Pro	Contra
http://www.zeit.de/campus/2013/s2/master-studium-entscheidung *Und für eine wissenschaftliche Karriere ist der Titel sowieso ein Muss, egal, in welchem Fach.* Unreferenced Qualitative	http://www.fr-online.de/berufsrundschau/abschluss-der-master-ist-kein-muss,4599958,17009424.html *Selbst wer in eine Führungsposition möchte, braucht den Master offenbar nicht zwingend. Nach der Umfrage können Bachelorabsolventen in 85 Prozent der Unternehmen jede Karriereposition erreichen.* Unreferenced Quantitative
http://www.zeit.de/campus/2010/s1/was-studieren-interview *Anzeige ZEIT Campus: Ein höheres Einstiegsgehalt garantiert der Master ja auch.* Unreferenced Qualitative	http://www.sueddeutsche.de/bildung/nach-dem-bachelor-warum-der-master-keine-entscheidenden-vorteile-bringt-1.1414318 *Für die meisten davon spielen die formalen Abschlüsse eine untergeordnete Rolle.* Unreferenced Qualitative
http://www.fr-online.de/berufsrundschau/abschluss-der-master-ist-kein-muss,4599958,17009424.html *Wer den Master macht, sollte sich auch inhaltlich*	http://www.fr-online.de/berufsrundschau/abschluss-der-master-ist-kein-muss,4599958,17009424.html *Der Master ist also kein Muss für eine erfolgreiche*

Figure 8.2: UI of the search results

9 Conclusion

This chapter presents the main contributions of this thesis and recommendations for future research.

9.1 Summary

In this thesis we present the conceptual design of a search engine which is used for obtaining the arguments regarding a specific topic and also develop the main components of such a system: a crawler, argument extraction and classification component and a front-end interface. The main contributions of this thesis are following:

- We develop a conceptual design of a search engine with the task to simplify the access to argumentation information concerning a specific topic.
- A focused crawler as well as a model for educational topics is created to perform crawling of topics in the educational domain.
- We introduce a new classification of arguments: Arguments by Polarity, Arguments by Argumentative Type and Arguments by Reference.
- A collection of 89 documents for 7 different controversial topics in educational domain is created. This collection is already used in a currently on-going Master's thesis of Roland Kluge *"Automatic Analysis of Arguments about Controversial Educational Topics in Web Documents"*.
- We perform an annotation study on the created collection with the goal to detect and classify arguments. As a result, to the best of our knowledge, we create the first labeled corpus for argumentation mining for German language in the educational domain, which also can be reused in further research. This corpus contains 487 arguments on 7 educational topics.
- To the best of our knowledge we also create the first automatic algorithm for boundary reconciliation. It can be used for creating "gold" boundaries for given boundaries of the annotators.
- We develop an easy to extend annotation tool for annotating arguments in a text. This tool can be also easily adapted for annotating different kind of textual entities and is also used in the annotation study of Roland Kluge for annotating premises and conclusions in an argument.
- We perform argument classification experiments for Arguments by Polarity, Arguments by Argumentative Type as well as for Arguments by Reference.
- To the best of our knowledge, we perform first argument extraction experiments for German language. This means that we establish a baseline in this field, which can be used in further research.
- For the argument extraction experiment we create a list of words indicative for argumentation in German language. The words in this list are partially translated from the analogous English words from the [KD93]. However, we also add our own words by analizing the arguments from our corpus.
- We implement a textual unit classification (classification of textual parts of the document, with context consideration) for the open-source framework DKPro TC. Previously it could only operate on document level.
- Finally, we create a prototype of a front-end interface for our system

9.2 Recommendations for future research

Main thesis points for future research:

- As a logical continuation of this work, a full working system based on the design we propose and using all the components we develop can be implemented.
- In this work we do not concentrate on the internal structure of the arguments. However, as it mentioned in Chapter 4 each argument can be split into different parts, i.e. premises and conclusion. The on-going Master's thesis of Roland Kluge *"Automatic Analysis of Arguments about Controversial Educational Topics in Web Documents"* takes this into consideration and uses the corpus we develop.
- In each experiment section in Chapter 7, we make suggestions, which can improve the performance of the particular experiment. These suggestion could be analyzed and implemented in further research on this topic.
- Finally, on the basis of the model we develop for the argument extraction and classification the cross-domain experiments could be performed in order to evaluate if the features with used are domain-dependent or not.

A List of seed URLs

http://www.zeit.de/gesellschaft/schule/index
http://www.spiegel.de/schulspiegel/
http://www.spiegel.de/unispiegel/
http://www.tagesspiegel.de/berlin/schule/
http://www.sueddeutsche.de/bildung
http://www.welt.de/themen/vorschule/
http://www.dw.de/themen/bildung/s-8009
http://www.focus.de/suche/bildung/
http://www.faz.net/aktuell/wissen/faktencheck/
http://www.berliner-zeitung.de/studium/10808902,10808902.html
http://www.bildung-news.com/
http://www.fr-online.de/bildung/24827914,24827914.html
http://www.derwesten.de/politik/campus-karriere/

Table A.1: List of seed URLs

B List of page URLs used in corpus

http://www.spiegel.de/schulspiegel/wissen/studie-jeder-sechste-lehrer-fuehlt-sich-gemobbt-a-866808.html
http://www.bildung-news.com/bildung-und-karriere/erfahrungsberichte/10-grunde-fur-eine-promotion/
http://www.spiegel.de/schulspiegel/wissen/neue-bildungsstudie-sitzenbleiben-ist-nutzlos-und-teuer-a-646709.html
http://www.sueddeutsche.de/bildung/niedersachsen-will-sitzenbleiben-abschaffen-aus-fuer-die-unruehmliche-ehrenrunde-1.1591350
http://www.welt.de/print/die_welt/politik/article109114444/Das-doppelte-Schul-Lottchen.html
http://www.dw.de/streit-ums-sitzenbleiben/a-16692803
http://www.zeit.de/studium/hochschule/2013-04/promotionen-anstieg-studentenzahlen
http://www.sueddeutsche.de/bildung/bildungssenator-in-hamburg-sitzenbleiben-nuetzt-nichts-und-verschwendet-viel-geld-1.1601356
http://www.erstenachhilfe.de/blog/Sitzenbleiben-abschaffen-Schueler-und-Studenten-sagen-Nein
http://www.spiegel.de/schulspiegel/ehrenrunden-debatte-deutsche-sind-fuer-das-sitzenbleiben-in-der-schule-a-181217.html
http://jetzt.sueddeutsche.de/texte/anzeigen/538752/Muss-ich-wirklich-noch-den-Master-machen
http://www.spiegel.de/schulspiegel/wissen/imageproblem-mehrheit-der-deutschen-haelt-lehrer-fuer-ueberfordert-und-unfaehig-a-615636.html
http://www.focus.de/schule/lehrerzimmer/schulpraxis/angst-wenn-schule-lehrer-krank-macht_aid_434812.html
http://www.sueddeutsche.de/bildung/inklusion-statt-foerderschule-wann-gemeinsames-lernen-sinnvoll-ist-1.1482320
http://www.focus.de/schule/schule/9000-lehrer-gehen-fuer-die-bildung-auf-die-strasse-lehrer-warnstreik-in-sachsen_aid_814848.html
http://daserste.ndr.de/guentherjauch/rueckblick/schulreform439.html
http://www.badische-zeitung.de/suedwest-1/auch-baden-wuerttemberg-ist-gegen-das-sitzenbleiben--69201353.html
http://www.bpb.de/apuz/32713/ueber-widersacher-der-inklusion-und-ihre-gegenreden-essay?p=all
http://www.sueddeutsche.de/karriere/phd-statt-promotion-auswandern-fuer-den-doktortitel-1.1029549
http://www.focus.de/schule/schule/bildungspolitik/tid-24802/abschied-auf-raten-sitzenbleiben-kommt-aus-der-mode_aid_684417.html
http://www.spiegel.de/schulspiegel/stress-im-klassenzimmer-jeder-dritte-lehrer-ist-ausgebrannt-a-244095.html
http://www.spiegel.de/schulspiegel/wissen/hessen-schueler-klagt-gegen-ungerechtigkeit-bei-turbo-abi-a-712926.html

Table B.1: List of corpus URLs. Part 1

http://www.spiegel.de/schulspiegel/wissen/bildungsforscher-sitzenbleiben-bringt-schuelern-kaum-vorteile-a-884286-druck.html
http://www.sueddeutsche.de/bildung/nach-dem-bachelor-warum-der-master-keine-entscheidenden-vorteile-bringt-1.1414318
http://www.zeit.de/campus/2013/s2/master-studium-entscheidung
http://www.zeit.de/campus/2010/s1/was-studieren-interview
http://www.spiegel.de/schulspiegel/wissen/behinderte-schueler-na-bitte-es-geht-doch-a-769821.html
http://www.heise.de/tp/artikel/38/38752/1.html
http://www.focus.de/schule/lernen/bildung-praemien-und-boni-fuer-gute-paedagogen_aid_459248.html
http://www.spiegel.de/unispiegel/jobundberuf/20-000-freie-stellen-deutschland-gehen-die-lehrer-aus-a-570627.html
http://www.tagesspiegel.de/meinung/jungen-und-maedchen-im-sportunterricht-getrennt-turnt-es-sich-besser/8035878.html
http://www.spiegel.de/schulspiegel/wissen/schulen-in-nrw-einige-gymnasien-wollen-turbo-abi-kippen-a-738375.html
http://www.spiegel.de/schulspiegel/wissen/lehrer-studie-weltweite-klagen-ueber-ruepel-schueler-a-630741-druck.html
http://www.myhandicap.de/behinderte-kinder-schule-inklusiv.html
http://www.faz.net/aktuell/wissen/faktencheck/faktencheck-hilft-das-sitzenbleiben-in-der-schule-12111532.html
http://blog.initiativgruppe.de/2013/04/10/sportunterricht-und-integration-jungen-und-madchen-zusammen-oder-getrennt/
http://www.welt.de/print/die_welt/finanzen/article114280562/Einige-Privilegien-wenig-Dank.html
http://www.spiegel.de/schulspiegel/sitzenbleiben-nichts-als-verplemperte-zeit-a-364198-druck.html
http://www.bildung-news.com/bildung-und-karriere/erfahrungsberichte/10-gute-grunde-nicht-zu-promovieren/
http://www.derwesten.de/staedte/luenen/jugend/getrennt-statt-im-team-im-sport-ein-volltreffer-id6785196.html
http://www.neues-deutschland.de/artikel/818437.streitfall-getrennter-sportunterricht.html
http://www.spiegel.de/schulspiegel/lachseminare-fuer-lehrer-lernziel-witzischkeit-a-193916-druck.html
http://www.welt.de/politik/deutschland/article114538623/Wie-Eltern-das-Projekt-Inklusion-torpedieren.html
http://www.spiegel.de/schulspiegel/wissen/teuer-sinnlos-frustrierend-weg-mit-der-ehrenrunde-a-551743.html
http://www.spiegel.de/schulspiegel/wissen/beamtenstatus-und-gehalt-ob-es-sich-lohnt-lehrer-zu-werden-a-877467-druck.html
http://www.derwesten.de/zeusmedienwelten/zeus/fuer-schueler/zeus-regional/gladbeck/geschlechtertrennung-im-sportunterricht-id3518467.html
http://www.zeit.de/2011/10/Ueberfluessige-Dissertationen
http://www.fr-online.de/berufsrundschau/abschluss-der-master-ist-kein-muss,4599958,17009424.html

Table B.2: List of corpus URLs. Part 2

http://www.welt.de/politik/deutschland/article13850739/Funktioniert-die-Schule-mit-der-vollen-Inklusion.html
http://www.spiegel.de/schulspiegel/ehrenrunde-sitzenbleiber-bringen-bessere-leistungen-a-316824-druck.html
http://www.ksta.de/job-und-karriere/promotion-mehr-geld-mit-doktortitel,20063080,22559908.html
http://www.change.org/de/Petitionen/wiedereinführung-des-g9-an-hamburger-gymnasien-mit-wahlfreiheit-zwischen-g8-und-g9
http://www.spiegel.de/schulspiegel/lehrer-lehnt-verbeamtung-ab-und-moechte-als-angestellter-arbeiten-a-877431-druck.html
http://www.jobvector.de/journal/bewerbung/soll_ich_promovieren/index_ger.html
http://www.spiegel.de/schulspiegel/wissen/faktencheck-wie-viel-arbeiten-lehrer-und-wie-viel-freizeit-haben-sie-a-874089-druck.html
http://www.zeit.de/2011/27/C-Interview-Prenzel
http://www.spiegel.de/schulspiegel/arme-lehrer-notfalls-gehe-ich-putzen-a-608995.html
http://www.haus-der-sprache.de/lektor.php/redaktion/lesen-karriere/promovieren_oder_nicht_was_bringt_der_doktorhut/
http://www.sueddeutsche.de/bildung/folgen-der-verkuerzten-schulzeit-setzen-sechs-1.1400905
http://www.welt.de/print/wams/vermischtes/article13556539/Ein-gutes-Abitur-braucht-seine-Zeit.html
http://www.spiegel.de/schulspiegel/lehrer-als-schulschwaenzer-protest-paedagogen-muessen-attest-vorlegen-a-247821.html
http://www.spiegel.de/unispiegel/studium/promovieren-doktortitel-kann-die-jobsuche-erschweren-a-843999.html
http://www.spiegel.de/schulspiegel/laut-umfrage-halten-deutsche-schueler-das-sitzenbleiben-fuer-richtig-a-887150-druck.html
http://www.christophburger.de/?p=1180
http://www.spiegel.de/unispiegel/jobundberuf/promotion-was-tun-dr-arbeitslos-a-252315.html
http://www.spiegel.de/schulspiegel/turbo-abiturienten-nach-klasse-12-wir-versuchskaninchen-a-781215.html
http://www.rbb-online.de/politik/beitrag/2013/07/maedchen_und_jungen_duerfen_getrennt_unterrichtet_werden.html
http://www.spiegel.de/schulspiegel/deutschlands-lehrer-raus-aus-der-schmollecke-a-347012.html
http://www.ismail-tipi.de/inhalte/2/aktuelles/35355/getrennter-sportunterricht-schadet-der-integration-und-ist-eine-steilvorlage-fuer-extremisten/index.html
http://www.spiegel.de/schulspiegel/wissen/g8-eltern-lehnen-turbo-abitur-ab-a-854096.html
http://www.welt.de/politik/deutschland/article114159103/Deutsche-Schueler-wollen-das-Sitzenbleiben-retten.html?config=print
http://www.faz.net/aktuell/beruf-chance/campus/karriere-persoenlichkeit-statt-promotion-1407397.html
http://www.welt.de/geschichte/article113734891/Viele-Sitzenbleiber-machten-doch-noch-Karriere.html

Table B.3: List of corpus URLs. Part 3

http://abi.de/studium/studiengaenge/weiterfuehrende/master09530.htm?zg=schueler
http://www.npd-brandenburg.de/inklusion---nicht-nur-fur-die-schuler-schlecht
http://www.spiegel.de/schulspiegel/wissen/kess-studie-zu-g8-und-g9-acht-jahre-gymnasium-reichen-aus-a-869483.html
http://www.ingenieur.de/Arbeit-Beruf/Ausbildung-Studium/Der-Master-lohnt-fuer-erfahrene-Kollegen
http://www.focus.de/schule/schule/recht/schule-sitzenbleiben-wird-abgeschafft_aid_295316.html
http://www.welt.de/welt_print/wissen/article4285459/Lohnt-sich-eine-Doktorarbeit.html
http://www.spiegel.de/schulspiegel/leben/schueler-berichten-ueber-sitzenbleiben-a-899607.html
http://www.sueddeutsche.de/bildung/debatte-um-gymnasialreform-mehr-zeit-weniger-stress-1.1301724
http://www.spiegel.de/schulspiegel/wissen/g9-jetzt-hamburger-eltern-starten-ini-fuer-neunjaehriges-gymnasium-a-900009.html
http://www.berliner-zeitung.de/berlin/getrennter-sportunterricht-maedchen-muessen-auf-den-schwebebalken,10809148,23868898.html
http://www.spiegel.de/schulspiegel/leben/cyber-mobbing-gegen-lehrer-pornomontagen-und-hinrichtungsvideos-a-488062-druck.html
http://www.welt.de/regionales/frankfurt/article111169857/Wahlfreiheit-bei-G8-G9-traegt-zu-Unruhe-bei.html
http://www.tagesspiegel.de/wissen/bildungsforschung-foerdern-statt-frustrieren/1593774.html
http://www.welt.de/politik/deutschland/article106221096/Disziplinlose-Schueler-ueberfordern-deutsche-Lehrer.html
http://www.spiegel.de/schulspiegel/lehrer-arbeitszeit-keine-fleisskaertchen-fuer-paedagogen-a-219833.html
http://www.focus.de/schule/schule/unterricht/inklusion/inklusion-eine-schule-fuer-alle_aid_684442.html

Table B.4: List of corpus URLs. Part 4

List of Figures

List of Tables

Bibliography

[AB13] Hector Martinez Alonso and Nuria Bel. Annotation of regular polysemy and underspecification. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 725–730, 2013.

[AP08] Ron Artstein and Massimo Poesio. Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596, 2008.

[AZ12] CharuC. Aggarwal and ChengXiang Zhai. A survey of text classification algorithms. In Charu C. Aggarwal and ChengXiang Zhai, editors, *Mining Text Data*, pages 163–222. Springer US, 2012.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, April 1998.

[BPM09] Sotiris Batsakis, Euripides G. M. Petrakis, and Evangelos Milios. Improving the performance of focused web crawlers. *Data Knowl. Eng.*, 68(10):1001–1013, October 2009.

[Bre01] Leo Breiman. Random forests. *Mach. Learn.*, 45(1):5–32, October 2001.

[Coh60] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

[DF82] Mark Davies and Joseph L. Fleiss. Measuring Agreement for Multinomial Data. *Biometrics*, 38(4):1047–1051, November 1982.

[EGR13] Nicolai Erbs, Iryna Gurevych, and Marc Rittberger. Bringing order to digital libraries: From keyphrase extraction to index term assignment. *D-Lib Magazine*, 19(9/10):1–16, September 2013.

[Fle71] Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, November 1971.

[For03] George Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289–1305, March 2003.

[Fou13] Chris Fournier. Evaluating Text Segmentation using Boundary Edit Distance. In *Proceedings of 51st Annual Meeting of the Association for Computational Linguistics*, page to appear, Stroudsburg, PA, USA, 2013. Association for Computational Linguistics.

[Gan12] Vaishali Ganganwar. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2:42–47, 2012.

[HCL03] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003.

[Hea97] Marti A. Hearst. Texttiling: segmenting text into multi-paragraph subtopic passages. *Comput. Linguist.*, 23(1):33–64, March 1997.

[HFH$^+$09] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[HFL$^+$12] Constantin Houy, Peter Fettke, Peter Loos, Iris Speiser, Maximilian Herberger, Alfred Gass, and Ulrich Nortmann. Argumentum - towards computer-supported analysis, retrieval and synthesis of argumentation structures in humanities using the example of jurisprudence. In Stefan Wölfl, editor, *KI-2012: Poster and Demo Track. German Conference on Artificial Intelligence (KI-12), 35th, September 24, Saarbrücken, Germany*, pages 30–33. DFKI, 9 2012.

[HGS10] Hua Huang, Shu Gao, and Chaojie Shao. Distributed search engine design and implementation based on lucene. In *Computer Design and Applications (ICCDA), 2010 International Conference on*, volume 4, pages V4–331–V4–334, 2010.

[HMY$^+$11] Hong-Wei Hao, Cui-Xia Mu, Xu-Cheng Yin, Shen Li, and Zhi-Bin Wang. An improved topic relevance algorithm for focused crawling. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 850–855, 2011.

[HNFL13] Constantin Houy, Tim Niesen, Peter Fettke, and Peter Loos. Towards automated identification and analysis of argumentation structures in the decision corpus of the German Federal Constitutional Court. *2013 7th IEEE International Conference on Digital Ecosystems and Technologies (DEST)*, pages 72–77, July 2013.

[JP13] V.S. Jagtap and Karishma Pawar. Analysis of different approaches to sentence-level sentiment classification. *International Journal of Scientific Engineering and Technology*, 2(3):164–170, 2013.

[KD93] Alistair Knott and Robert Dale. Using linguistic phenomena to motivate a set of rhetorical relations. Technical report, Discourse Processes, 1993.

[KS12] Anna Kazantseva and Stan Szpakowicz. Topical segmentation: a study of human performance and a new measure of quality. In *HLT-NAACL*, pages 211–220. The Association for Computational Linguistics, 2012.

[KV13] Mukesh Kumar and Renu Vig. Focused crawling based upon tf-idf semantics and hub score learning. *Journal of Emerging Technologies in Web Intelligence*, 5(1), 2013.

[LK77] J.R. Landis and G.G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 32(1):159–174, March 1977.

[MBPR07] Marie-Francine Moens, Erik Boiy, Raquel Mochales Palau, and Chris Reed. Automatic detection of arguments in legal texts. In *Proceedings of the 11th international conference on Artificial intelligence and law*, ICAIL '07, pages 225–230, New York, NY, USA, 2007. ACM.

[MN98] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *IN AAAI-98 WORKSHOP ON LEARNING FOR TEXT CATEGORIZATION*, pages 41–48. AAAI Press, 1998.

[Mom12] Saeedeh Momtazi. Fine-grained german sentiment analysis on social media. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *LREC*, pages 1215–1220. European Language Resources Association (ELRA), 2012.

[NOPF10] David Nettleton, A. Orriols-Puig, and A. Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, 33:275–306, 04/2010 2010.

[Pla98] John C. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical report, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING, 1998.

[PM09] R M Palau and M F Moens. Argumentation mining: the detection, classification and structure of arguments in text. 2009.

[PNPH08] Mari-Sanna Paukkeri, Ilari T. Nieminen, Matti Pöllä, and Timo Honkela. A language-independent approach to keyphrase extraction and evaluation. In *Coling 2008: Companion volume: Posters*, pages 83–86, Manchester, UK, August 2008. Coling 2008 Organizing Committee.

[Pow07] David M. W. Powers. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Technical Report SIE-07-001, School of Informatics and Engineering, Flinders University, Adelaide, Australia, 2007.

[Qui96] J. R. Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[RR04] Chris Reed and Glenn Rowe. Araucaria: Software for argument analysis, diagramming and representation. *International Journal of AI Tools*, 14:961–980, 2004.

[SPW$^+$13] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, October 2013. Association for Computational Linguistics.

[Teu99] Simone Teufel. Argumentative Zoning : Information Extraction from Scientific Text University of Edinburgh. 1999.

[TNGST10] Nguyen Thai-Nghe, Zeno Gantner, and Lars Schmidt-Thieme. Cost-sensitive learning methods for imbalanced data. In *IJCNN*, pages 1–8. IEEE, 2010.

[WK09] Michael Wiegand and Dietrich Klakow. The role of knowledge-based features in polarity classification at sentence level. In H. Chad Lane and Hans W. Guesgen, editors, *FLAIRS Conference*. AAAI Press, 2009.

[WRM08] D. Walton, C. Reed, and F. Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

[WvL08] U. Weinreich and E. von Lindern. *Praxisbuch Kundenbefragungen: repräsentative Stichproben auswählen ; relevante Fragen stellen ; Ergebnisse richtig interpretieren*. Werben & Verkaufen. mi, 2008.

[WWH05] Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 347–354, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

[ZQDS03] Junlin Zhang, Weimin Qu, Lin Du, and Yufang Sun. A framework for domain-specific search engine: design pattern perspective. In *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, volume 4, pages 3881–3886 vol.4, 2003.