

Summarization and Topic Extraction in Argumentation Graphs

Master's Thesis

by

Olga Batiukova

born in

Kertsch, Ukraine

submitted to

Professorship for Computer Networks

Prof. Dr. Martin Mauve

Heinrich-Heine-University Düsseldorf

September 2020

Supervisor:

Dr. Christian Meter

Abstract

Within this master's thesis, an extension software for the dialog-based argumentation platform *D-BAS* has been developed. The proposed tool comprises a topic extractor, a summary generator and a keyword extractor. The main purpose of this tool is to allow users to extract the most significant information from the *D-BAS* discussions.

Subsequently, the accuracy of the results generated by the proposed extension tool has been assessed by means of the human evaluation as well as statistical metrics. The obtained results suggest that this software service is able to cover the average needs of the users. However, several enhancements should be performed in the future in order to increase the accuracy of the output data.

Acknowledgments

First of all, I would like to thank my supervisor, Dr. Christian Meter, for his guidance and valuable pieces of advice he gave me. I am very grateful for his patience and readiness to answer all my questions.

Additionally, I would like to express my appreciation to Marc Feger and Jan Lukas Steinmann for providing me with additional sources of information and giving me new ideas.

Special thanks to everyone who took part in the evaluation process and responded to all the numerous tasks within the questionnaires. I know how tiresome it was, thank you so much!

I am also very grateful to "the most awesome" friend, Alexander Diez, for proof-reading this thesis and cheering me up throughout the whole process.

And finally, from the bottom of my heart I wish to express my gratefulness to my family. My dear Mom and Dad, thank you for your love, your support, your care and for believing in me! I love you soooooo much!!!

Contents

1. Introduction	1
1.1. Motivation	1
1.2. Structure of the Thesis	2
2. Preliminaries	3
3. Argumentation Systems	5
3.1. Discussion Forums	6
3.2. Pro and Contra Lists	6
3.3. Argumentation Maps	7
3.4. Argumentation Systems	7
3.4.1. Dialog-Based Argumentation Systems	7
4. Topic Modeling, Automatic Text Summarization and Keywords Extraction	13
4.1. Preprocessing	13
4.1.1. Tokenization	14
4.1.2. Lowercasing	14
4.1.3. Stemming	14
4.1.4. Lemmatization	15
4.1.5. Stopword removal	15
4.2. Topic Modeling	16
4.2.1. Matrix Factorization Approach	18
4.2.2. Generative Approach	19
4.2.3. Input Matrix: TF vs. TF-IDF	20
4.2.4. LSA: Latent Semantic Analysis	21
4.2.5. pLSA: Probabilistic Latent Semantic Analysis	21
4.2.6. LDA: Latent Dirichlet Allocation	23
4.2.7. NMF: Non-negative Matrix Factorization	24

4.2.8.	Topic Modeling Tools	25
4.3.	Automatic Text Summarization	25
4.3.1.	Position-based Approach	27
4.3.2.	Term-based Approaches	27
4.3.3.	Graph-based Approach: TextRank	28
4.3.4.	Machine Learning Approach	30
4.4.	Automatic Keywords Extraction	30
4.4.1.	Term-based Approaches	31
4.4.2.	Term-based Approaches: RAKE	31
4.4.3.	Graph-based Approach: TextRank	33
4.4.4.	Machine Learning Approach	33
5.	Implementation	35
5.1.	General Project Structure	35
5.2.	Topic Modeling: <i>topics</i> app	38
5.2.1.	Parameters	39
5.2.2.	Use Cases	40
5.2.3.	Data preprocessing	42
5.2.4.	Pre-trained Models: External Datasets	42
5.2.5.	Topic Model	44
5.2.6.	API	45
5.3.	Text Summarization: <i>summary</i> app	46
5.3.1.	Parameters	47
5.3.2.	Use Cases	48
5.3.3.	Input Text Preparation	50
5.3.4.	Summarization	51
5.3.5.	API	52
5.4.	Keywords Extraction: <i>keywords</i> app	53
5.4.1.	Parameters	54
5.4.2.	Use Cases	55
5.4.3.	Keyword Extraction	57
5.4.4.	API	58
6.	Evaluation	61
6.1.	Topic Modeling	61
6.1.1.	Evaluation Setup	62

6.1.2.	Topics with Pre-trained Models	63
6.1.3.	Topics without Pre-trained Models	66
6.2.	Text Summarization	70
6.2.1.	Evaluation Setup	70
6.2.2.	Human Evaluation	71
6.2.3.	ROUGE Scores	73
6.3.	Keyword extraction	78
6.3.1.	Evaluation Setup	78
6.3.2.	Human Evaluation	79
6.3.3.	ROUGE Scores	81
7.	Conclusion	83
	Appendices	85
A.	50 pre-trained topics (<i>gensim LDA</i> English model)	87
B.	50 pre-trained topics (<i>sklearn LDA</i> English model)	93
C.	50 pre-trained topics (<i>gensim LDA</i> German model)	99
D.	50 pre-trained topics (<i>sklearn LDA</i> German model)	105
E.	Generated Summaries	111
E.1.	English <i>gensim</i> Summary	111
E.1.1.	Without considering a position structure	111
E.1.2.	Considering a position structure	111
E.2.	English TextRank Summary	112
E.2.1.	Without considering a position structure	112
E.2.2.	Considering a position structure	112
E.3.	English TF Summary	113
E.3.1.	Without considering a position structure	113
E.3.2.	Considering a position structure	113
E.4.	English TF-IDF Summary	113
E.4.1.	Without considering a position structure	113
E.4.2.	Considering a position structure	114

E.5. German gensim Summary	114
E.5.1. Without considering a position structure	114
E.5.2. Considering a position structure	114
E.6. German TextRank Summary	115
E.6.1. Without considering a position structure	115
E.6.2. Considering a position structure	115
E.7. German TF Summary	116
E.7.1. Considering a position structure	116
E.7.2. Considering a position structure	116
E.8. German TF-IDF Summary	116
E.8.1. Without considering a position structure	116
E.8.2. Considering a position structure	117
E.9. Reference Summaries	118
F. Generated Keywords	121
List of Figures	123
List of Tables	127
Bibliography	129

Chapter 1.

Introduction

Progress defines the way of living in all possible aspects of a human life. It made it easier for people to travel by linking cities and countries together. Progress changed the way how more feasible buildings are erected. It allowed companies to more effectively do their business and allowed economies to grow. And surely, it changed the way how people communicate. The days when people needed to wait for a letter to arrive are gone. The Internet has changed the style, speed and quality of the communication. Nowadays, it takes a couple of minutes if not seconds to reply to someone's message or to express one's opinion, and numerous applications and frameworks were created to enable online discussions.

In Section 1.1, the motivation for this master's thesis is given. Section 1.2 gives a brief overview of the structure of this thesis.

1.1. Motivation

Though, such an accessibility has numerous advantages, it also creates an enormous amount of information which users are encountered with. While being engaged in online discussions, participants often do not know what it started with, what other users already said or what the whole discussion is about.

Within the framework of this master's thesis, an extension tool for the Dialog-Based Argumentation System *D-BAS* is developed that can allow participants to get an overview of the

discussions, as well as their parts, by summarizing the discussed opinions. This tool can also enable users to keep track of the main topics of the discussion and retrieve keywords for their parts.

1.2. Structure of the Thesis

This master's thesis is structured as follows.

In Chapter 1, the motivation for this thesis, as well as its structure, are presented.

Chapter 2 gives an overview of the main argumentation system approaches and highlights the main features of the *D-BAS* system.

Chapter 3 provides definitions of the main concepts which will be used throughout this master's thesis.

In Chapter 4, the main approaches and algorithms, used in the fields of topic modeling, text summarization, and keywords extraction are introduced.

An explanation of the decision-making process behind an implementation of the proposed tool is given in Chapter 5.

In Chapter 6, the results produced by the proposed tool are presented.

Finally, a short summary, as well as ideas for future work are presented in Chapter 7.

Chapter 2.

Preliminaries

This chapter provides definitions of the main concepts, which will be used throughout this thesis.

All the algorithms, presented in this thesis, deal with natural language texts. In order to perform any computational operation on an input text, this text should first be preprocessed. *Text Preprocessing* is a series of text cleaning and normalization methods, in which the input text is transformed in such a way, so that it is suitable for further application of the algorithms. In a more detailed way, this process is explained in Section 4.1.

Topic modeling is a suit of algorithms for automatic extraction of abstract (hidden) themes called *topics* from a collection of documents known as a *dataset*. This goal can be achieved by using unsupervised machine learning. Section 4.2 provides more information on this subject.

Unsupervised machine learning is a technique, where no predefined human-tagged data is used to train a model, which implies that a human supervision is reduced to its minimum.

Automatic Text Summarization is the process of reducing a size of an input text in such a way, so that a shorter output text, containing all the crucial points of the original text, can be produced. More information about text summarization is given in Section 4.3.

Automatic Keywords Extraction is a technique that identifies and extracts a set of the most significant terms and expressions from an input text. These keywords and key phrases can be later used for topic recognition, text summarization, etc. Section 4.4 gives detailed information concerning the automatic keyword extraction.

Chapter 3.

Argumentation Systems

Communication is a vital part of a human nature. It started with signs and sounds, evolved into speech, and was later extended with writing. And though the development of writing made it possible to send messages from one place to another, the process of exchanging information was still limited geographically. The development of computer technologies extended our world to an additional — digital — dimension, which provided a way of disconnecting people from their location and time and letting them communicate more easily. Information can now be acquired on the websites, and the exchange of opinions is performed via comments, forums, chats, etc.

The main reason why people share their opinion is because they want their voice to be heard. That is why online tools are a great way for them to participate in a social life. Online participation gives everyone a chance to influence an environment and living conditions without being bound to a particular place.

Such online tools speed up and simplify an opinion exchange. They allow people to take part in the decision-making processes because they are easily accessible for everyone. Such tools also reduce the number of people required to support the participation process and making it not only accessible, but also cost-effective.

This chapter presents the main approaches for development of online communication tools.

3.1. Discussion Forums

Online forum discussions are one of the oldest and mainly used forms of the online participation. Their goal is to encourage the engagement and interaction of participants. To start a discussion a user describes the problem. The exchange of information takes place through posting comments to the described problem or to comments of other participants.

Forums have a hierarchical structure, where each discussion is represented as an asynchronous thread. They can consist of several subforums, and each subforum can contain several threads. One of the best examples of discussion forums is *Reddit*, also known as the "front page of the Internet" [Red]. It is a massive collection of forums that people use to exchange content or comment on other people's posts.

The main advantages of discussion forums are the simplicity in use and their popularity. At the same time, they also have several disadvantages [Kle10]. Firstly, posts can be longer than one line of text. This increases an amount of information, some of which can be redundant. Secondly, a growing number of participants makes such tree-like structures as forums lose their transparency very quickly. It means that forums fail to scale. Thirdly, such systems contribute to the polarization of opinions, dividing opinions into sharply contrasting groups. Fourthly, forums do not reflect the attitude of participants towards the messages, which they reply to.

3.2. Pro and Contra Lists

Another approach for maintaining an online decision-making process is to use pro and contra lists. Such lists maintain an information exchange process by distinguishing between pro- and contra-opinions of participants which adds some structure to the discussion.

Such tools as *Kialo* [Kia], *ConsiderIt* [Kri+12] and *DebateHub* [DLB20] adopt the list approach. These tools also make it possible to rank the proposals. But pro and contra lists do not maintain the hierarchy of the entries. They also can not prevent the process of opinion polarization. Another important disadvantage of lists is that it is impossible for users to freely exchange arguments and counter arguments [KMM17].

3.3. Argumentation Maps

Argumentation mapping systems, such as *Carneades* [WG17], *Deliberatorium* [KI08], *bCisive* [Rea20], or *ArguNet* [SVB07], allow to create an argument map of an online discussion making it structured. Not only do these tools preserve the arguments of the users, they also preserve the relations between them. Online systems for argument mapping scale with an increasing amount of users. They also enable a free argument exchange. The main disadvantage of such tools is the complexity in user experience, since participants have to possess at least basic knowledge in formal argumentation. This makes argumentation maps to be expert tools rather than communication tools for common users.

3.4. Argumentation Systems

Argumentation systems are developed to reflect how human argumentation uses conflicting information to construct and analyze arguments. They guide the participants through a dialog process with the help of predefined rules. Argumentation systems model the argumentation in logic and identify pro and contra arguments relevant to an issue. One of the examples of the argumentation systems is *IBIS* — *Issue-Based Information System* [KR10].

3.4.1. Dialog-Based Argumentation Systems

Dialog-based argumentation systems are the next step in the development of efficient online participation tools. They combine all the advantages of argumentation mapping, such as structure, scalability, argument exchange, and add such an important feature as simplicity in use. With such a system, no prior knowledge is required for participants to take part in an online discussion.

One of the main strengths of dialog-based argumentation tools is the ability to lead a dialog between a system and a user. For example, such a dialog-based argumentation system as *D-BAS*, that will be described in more detail in the next section, suggests a user a single argument and a list of responses. After the user selected an appropriate response, the system suggests the user another argument that the user might be interested in. The choice of the

arguments is based on the data collected from previous users and the response of the current user. This information allows the system to accumulate knowledge, in order to perform more efficient and relevant argument selection for the next participants. Such an approach distinguishes dialog-based argumentation systems from techniques described in the previous sections.

D-BAS — A Dialog-Based Online Argumentation System

D-BAS [Kra+18] is a dialog-based online argumentation system, which was developed by the group of professor Martin Mauve at the University of Düsseldorf, and can be used for e-participation purposes. This system guides users through the argumentation process by proposing them one argument at a time and suggesting them possible responses to each argument, which were already used by previous users. Every participant can also add her own response, which can be later utilized by *D-BAS* for the next users. Such a dialog between the system and users has two main advantages. Firstly, it reduces a possible redundancy by reducing the amount of user arguments. Secondly, it eliminates a direct interaction between users making the system scale horizontally.

Though this approach manages to reduce the amount of information, some improvements are required to give users an overview over a complete discussion. One of these improvements is to develop a tool that performs topic modeling on the discussion data and provides an API for users to retrieve the results. This can allow users to understand what a particular discussion is about within a necessity to read it.

One more possible extension is an in-built text summarizer. It can internally reduce the amount of discussion statements to the most important ones and provide an API, so that the users can retrieve the results.

In more detail these improvements will be introduced in the subsequent chapters.

Argumentation Graph

Arguments in *D-BAS* are stored within a directed graph. An example of such a graph is shown in Figure 3.1.

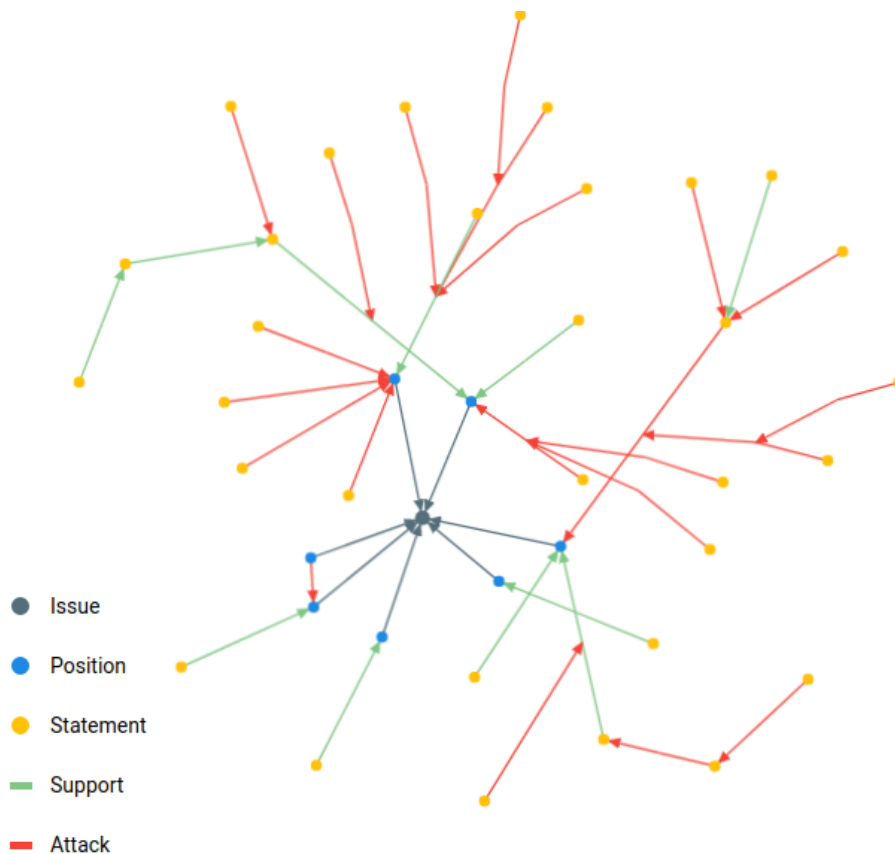


Figure 3.1.: Example graph in *D-BAS* [Net20].

Each argumentation is initiated by an *issue* that represent a topic of a discussion. The issue is represented by a grey circle on the graph in Figure 3.1. Issues contain all the necessary information about discussions: their titles, short descriptions, etc.

Statements, marked with yellow circles in Figure 3.1, are basic reusable units of an online discussion. An example of statements is shown in Figure 3.2. Each *argument* consists of one or more statements. Users can utilize the same arguments for other issues by connecting a statement sub-graph from one issue with the graph of another topic. Figure 3.3 illustrates two arguments that share the same conclusion. A directed edge between a premise-statement and a conclusion-statement, marked in orange, represents an argument. The same applies to the statements marked in black. Arguments reflect an attitude and represent either a support or an attack. The types of attacks are explained in Subsection 3.4.1.

Positions, represented as blue circles on the graph in Figure 3.1, are special types of statements, which recommend users to take some action. They have a neutral meaning and can

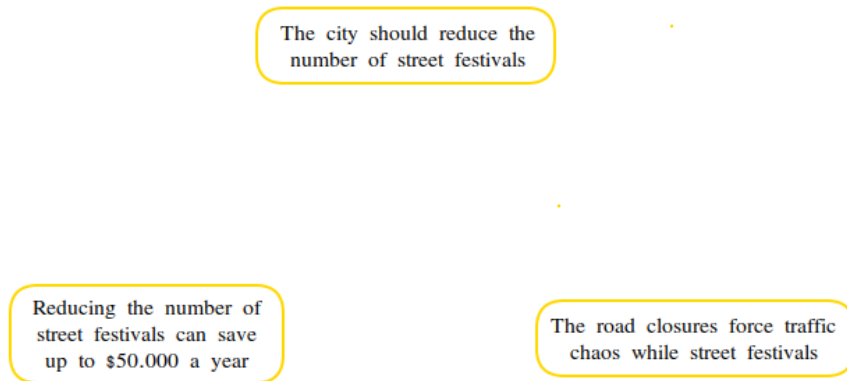


Figure 3.2.: Example of statements.

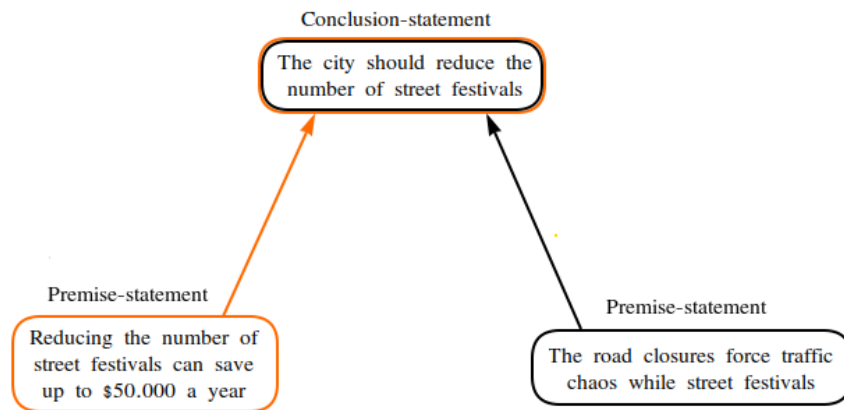


Figure 3.3.: Example of an argument.

not be utilized for other issues. Positions are supposed to enable the branching within a discussion creating different development directions. By supporting a particular position, the user follows the graph in this particular direction. In Figure 3.4, a position is given within a blue box.

Relations

As every dialog-based system, *D-BAS* tries to gather user's feedback to a given argument. The system gives its users a chance to choose from different sets of answers, which allows the system to make sure that the feedback is precise. One of the peculiarities of *D-BAS* is that it stores information about users attitudes in order to different arguments. For this purpose,

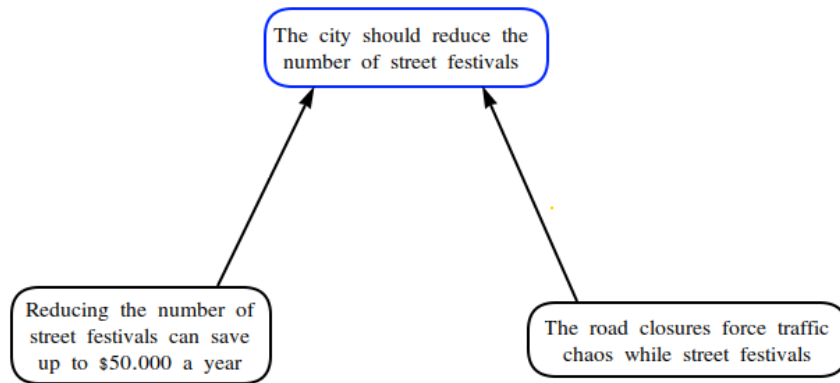


Figure 3.4.: Example of a position (the statement in the blue box).

the attacks from argumentation theory were adopted.

The information about user’s attitude is stored as relations between the graph nodes. Figure 3.5 shows a piece of an example discussion, where the users’ statements are marked with capital letters from *A* to *E* and relations between them are represented as colored arrows marked with numbers.

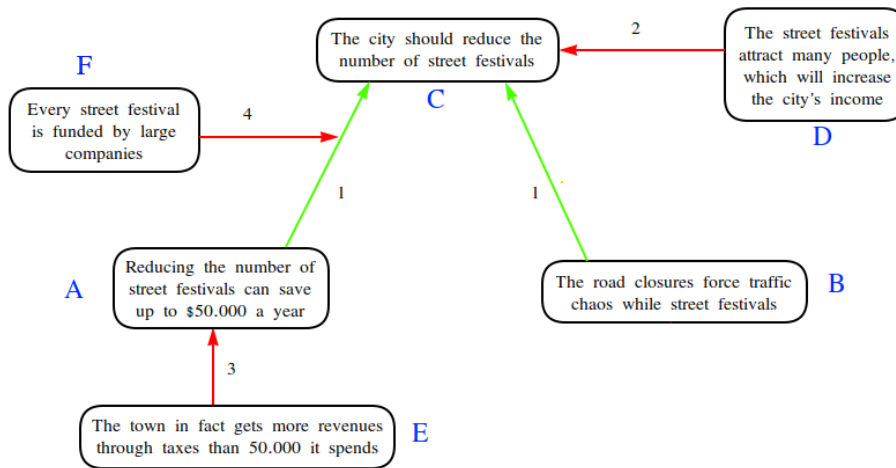


Figure 3.5.: Example of various attack types.

A *support* relation reflects a positive attitude towards an argument, which means that the user accept the premise and the conclusion. In Figure 3.5, both statements *A* and *B* support *C* because both *A* and *B* have a positive attitude to the premise of the *C*. This relation is represented by green arrows.

An *undermine* occurs if an argument with an opposite attitude towards another argument attacks its premise. In Figure 3.5, 3 rejects the premise of A and undermines 1.

An *undercut* reflects the situation, in which one argument attacks another argument by expressing the negative attitude, implying that the correct premise of the attacked argument and the conclusion are not linked. In other words, the argument accepts the premise but disagrees that this leads to accepting the conclusion. In Figure 3.5, the argument 4 undercuts the argument 1 by attacking it.

A *rebut* relation takes place when the arguments have the same conclusion, but express opposite attitudes. In other words, the argument accepts the premise but states that there is a stronger argument, which leads to rejecting the conclusion. In Figure 3.5, 2 rebuts 1 on C.

Chapter 4.

Topic Modeling, Automatic Text Summarization and Keywords Extraction

The first step towards an implementation of the proposed tool is to understand, what *Topic Modeling*, *Text Summarization*, and *Keywords Extraction* are. This chapter provides an overview of the main techniques, which proved to be the most successful in order to achieve good results.

4.1. Preprocessing

The input text preprocessing has a crucial meaning and affects the output results drastically [GK14].

The most important stages in the text preprocessing are: *tokenization*, *lowercasing*, *stemming*, *lemmatization*, and *stopwords removal*. All these techniques help to reduce the dimensionality of the input data by removing the noisy information.

4.1.1. Tokenization

Tokenization is the process of splitting a parsed text into units called *tokens*, which can be words, phrases or some other meaningful units. The list of tokens is then fed into the further preprocessing. Tokenization step is crucial for further processes, such as removal of unwanted units or construction of n-grams.

4.1.2. Lowercasing

Lowercasing is the process of mapping words with capital letters to the same lowercase words. This step makes sure that such words as, for example, *Cat* and *cat* are treated in the same way. The importance of the lowercasing increases, if the input data comes from the Internet (posts, comments, etc.), since a lot of words might be misspelled or their writing conventions may differ like in the example below.

$$\left. \begin{array}{l} D - BAS \\ d - BAS \\ d - bas \end{array} \right\} \rightarrow d - bas$$

Thus, the lowercasing is a helpful technique and leads to better results for the tasks, in which preserving capitalization is not important, it can cause a confusion in the tasks, in which capitalization matters. In the next step after lowercasing, the tokens have to be normalized by applying stemming and/or lemmatization techniques.

4.1.3. Stemming

One of the normalization techniques is *stemming*. It is the process of reducing inflection of a word to its base form or stem. Through stemming it is possible to define units from the same context. The stem of a word might not always be a real root, since the process of stemming chops suffixes or endings off following the artificially defined rules and not the linguistically defined ones. However, it helps to reduce the complexity of the input data.

For instance, different forms of the word *change* can be stemmed in the way shown in the example below.

$$\left. \begin{array}{l} \textit{change} \\ \textit{changes} \\ \textit{changing} \\ \textit{changed} \\ \textit{changer} \end{array} \right\} \rightarrow \textit{chang}$$

4.1.4. Lemmatization

Lemmatization is also a process of normalization, which is similar to stemming, but which makes an attempt to transform a word to its actual grammatical root. Two examples of lemmatizing different forms of the verb *to be* and different singular and plural forms for the word “goose” are shown below.

$$\left. \begin{array}{l} \textit{is} \\ \textit{are} \\ \textit{was} \end{array} \right\} \rightarrow \textit{be} \quad \left. \begin{array}{l} \textit{goose} \\ \textit{geese} \end{array} \right\} \rightarrow \textit{goose}$$

4.1.5. Stopword removal

Some words are encountered in the input text very often but do not contribute to acquiring any kind of information. For example, conjunctions or prepositions are used to connect words, but as a single unit they carry no useful information. Such words are called *stopwords*. They can be found in any language and should be removed during the preprocessing step. Additionally, it is worth of mentioning that the stopwords lists can differ depending on an input text and a task.

4.2. Topic Modeling

Topic models provide their users with topics, which represent a collection of words and phrases clustered in such a way, so that they preserve a high level of similarity within a cluster, but also retain a high level of distinction between several clusters. In other words, terms that share a thematic similarity can be collected into one group, but such groups differ from each other by their meanings.

Consider Figure 4.1, in which an example piece of text in a box below is given. This text consists of words marked with different colors depending on their relation between each other. Above this piece of text four topics are presented. The words listed in these topics are the most representative ones. The colored words from the text below can be grouped and assigned to one of these four topics: words marked with red definitely have something to do with “Arts“ topic, words in pink relate to “Education“ topic, etc.

“Arts”	“Budgets”	“Children”	“Education”
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give \$1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. “Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services,” Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center’s share will be \$200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive \$400,000 each. The Juilliard School, where music and the performing arts are taught, will get \$250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual \$100,000 donation, too.

Figure 4.1.: An example text document. Related words of the same color can be grouped into one topic. Original image from [BNJ03].

An additional feature, which topic models provide their users with, is an association of doc-

uments to those topics. This association can be viewed as a simplex. Figure 4.2 depicts a possible simplified version of a simplex for only three topics from the example above.

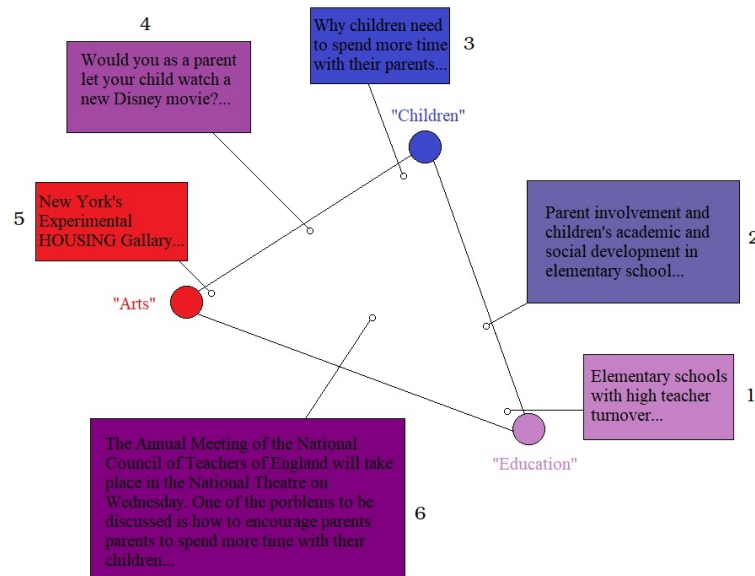


Figure 4.2.: A possible association of documents to topics from Figure 4.1. For simplicity, only three topics were chosen.

Each colored box in Figure 4.2 represents a document. The color of each document represents, to which extent this document relates to one of the three topics: “Arts“ (red), “Children“ (blue), “Education“ (pink). For instance, *Document 1* has a strong relation to the topic “Education“, but a weak relation to the topic “Children“. *Document 3* has, in its turn, a strong relation to the topic “Children“, but no relation to the topic “Education“. *Document 2* combines topics “Children“ and “Education“ prevailing towards the latter. In contrast to all other five documents, *Document 6* combines all three topics in almost equal proportions.

To sum the previous examples up, the following conclusion can be drawn. All topic models share a common hypothesis:

- documents consist of a combination of topics, and
- topics consist of a combination of words.

Thus, topic modeling can be used not only to retrieve topics from a given dataset, but also to classify, which topics a document consists of depending on a trained model.

The application fields of topic modeling vary. The most obvious use of this technique is text categorization [BCD10], [HMK12]. In bio-informatics, topic modeling can be applied to interpret bio-medical information [Liu+16], [ZZC14]. In Chemistry, topic modeling is used to assign large molecule sets to topics, in order to inspect the dependencies between them [Sch+17]. In sentiment analysis, topic modeling helps to examine a huge amount of user-generated blog posts, forum comments, social media messages [RCL16], [NS15]. Topic models can also solve problems in the area of object recognition by means of “visual” words and their locations on images [LZ10].

4.2.1. Matrix Factorization Approach

One way of thinking about topic modeling is in terms of a *Matrix Factorization Approach*. An entire dataset can be encoded as a *Term x Document* matrix X . Each entry in this matrix corresponds to the amount of times each term was seen in each document. In Figure 4.3, this matrix is shown on the left.

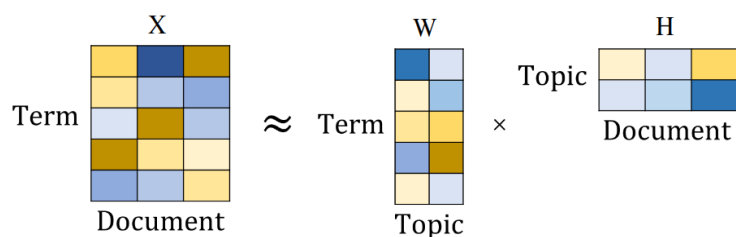


Figure 4.3.: A visualization of a dataset as a matrix factorization problem [Suh+16].

This matrix can be approximated by a product of two smaller matrices. For this, consider the right side of the same example. The *Term x Topic* matrix W represents an affinity of each word in a topic. This corresponds to the distribution of the words over each topic. The *Topic x Document* matrix H encodes, how likely a topic is to be seen in a document. This corresponds to the allocation simplex, explained in Figure 4.2.

4.2.2. Generative Approach

An alternative way of solving topic modeling tasks is called a *Generative Approach*. This approach implies the existence of a random generative process. It means that a model makes assumptions, on how documents were composed, by computing probability distributions [BCD10]. While computing these probabilities, *latent* variables are used. These latent variables are essential to the data, but can not be directly measured, since they are not original features of the dataset (they are hidden). To uncover the meaning of these variables, a *probabilistic inference* is used. Probabilistic inference derives a “[. . .] probability of one or more random variables[.]” [Gro].

As an example, consider Figure 4.4. It displays a generative process for an LDA model, which is described later in Subsection 4.2.6.

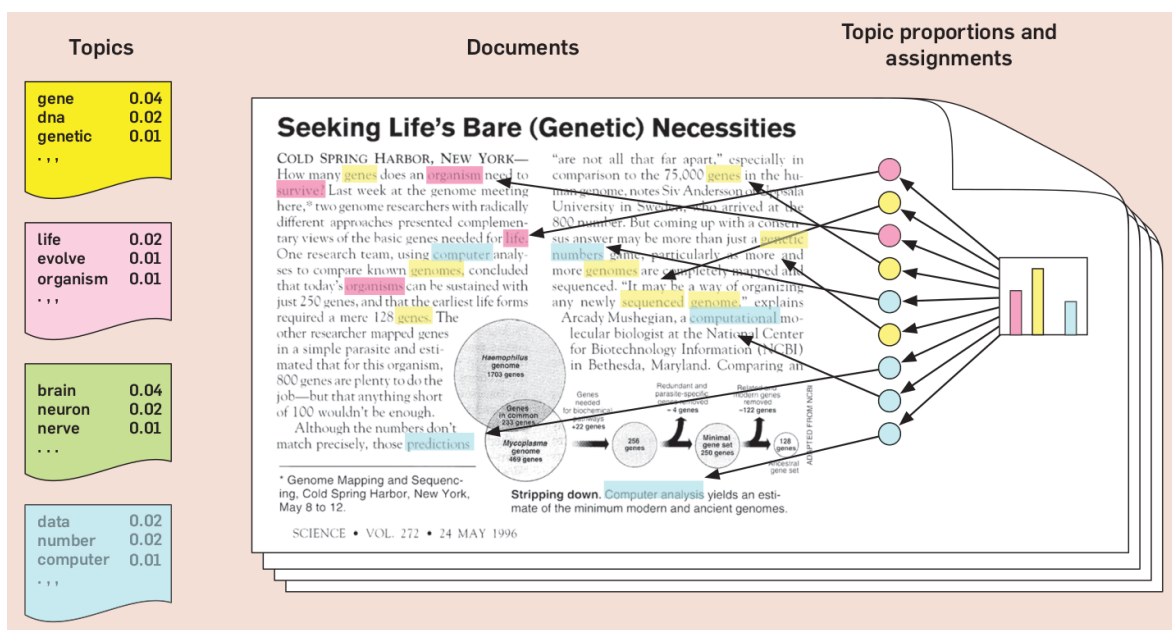


Figure 4.4.: An example of a generative process for an LDA model. Graphic taken from: [BCD10].

According to the generative process, which lies in the core of the LDA model, new documents are created in the following way:

- determine number of words in a document,
- choose a topic mixture for the document over a fixed set of topics,

- generate the words in the document by first picking a topic based on the document's distribution, and then pick a word based on the topic's distribution

4.2.3. Input Matrix: TF vs. TF-IDF

All previously explained approaches assume that a given dataset has to be converted into a *Term x Document* matrix for further calculations. To achieve better results, the entries of this matrix are usually weighted. There are numerous approaches for weighting words [SB88]. Two most commonly used ones are *Term Frequency (TF)* and *Term Frequency - Inverse Document Frequency (TF-IDF)*.

Term Frequency Weighting is based on the assumption that the importance of a term is proportional to the number of times it appears in the document. Consider Equation 4.1 and the following notation:

- w_i is the weight of term i ,
- tf_i is the raw frequency of term i in a document

$$w_i = tf_i \tag{4.1}$$

TF denotes a raw frequency of each term. This means that a number of times each term appears in a document corresponds to the weight of this word. This approach is straightforward, but it does not take into account the frequency of the term in the whole dataset.

Term Frequency - Inverse Document Frequency approach eliminates this flaw. It assumes that if terms with high frequency occur in a vast amount of documents, its value decreases. But if words with high frequency appear only in a small amount of documents, then they are more representative. Consider Equation 4.2 with the following notation:

- w_i is the weight of term i ,
- tf_i is the raw frequency of term i in a document,

- N is the total number of documents,
- N_i is the number of documents that contain term i ,

$$w_i = tf_i \cdot \log\left(\frac{N}{N_i}\right) \quad (4.2)$$

TF-IDF first computes the term frequency of each term and then normalizes it with the inverse document frequency. In this way, not the frequency of the word is measured but its relevance. As a result, “meaningless“ words like, for instance, prepositions can be taken out of the equation, and only distinct terms are considered.

Now, when the basic concepts of topic modeling are explained, the main topic modeling algorithms can be introduced.

4.2.4. LSA: Latent Semantic Analysis

Latent Semantic Analysis (LSA), or *Latent Semantic Indexing (LSI)*, is one of the basic techniques in topic modeling. This model was proposed by Jerome Bellegarda in 2005 [Bel05].

LSA consists of two steps. In the first step, a dataset is turned into a *Term x Document* matrix, explained in the previous section. Since this matrix has a high sparsity, a *Singular Value Decomposition (SVD)* is used in the second step. The crux of this approach is to remove all the noisy information and to reduce the dimensionality, so that only valuable information remains. Though, in large datasets this technique can achieve a significant compression, this approach fails to deal with polysemy and synonymy [BNJ03].

4.2.5. pLSA: Probabilistic Latent Semantic Analysis

Probabilistic Latent Semantic Analysis (pLSA), or *Probabilistic Latent Semantic Indexing (pLSI)*, is an improved version of LSA. This model was introduced by Hofmann in 1999

[Hof99]. pLSA makes use of a generative probabilistic approach instead of SVD. This model predicts with which probabilities words and topics were drawn to generate documents within a dataset. Figure 4.5 displays a plate notation of the model, in which:

- M is a total number of documents,
- N is a total number of words in a document,
- w is a single word,
- d is a document label,
- z is a latent topic variable.

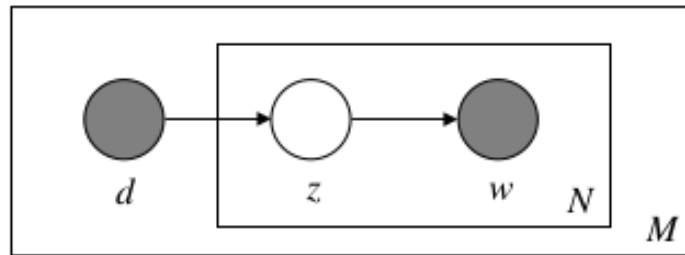


Figure 4.5.: A plate notation of a pLSA model [BNJ03].

The parameters w and d are in grey circles, which indicates that these are the only observable parameters. The latent topic z can be found within the document d with some probability $p(z|d)$. The word w_n is drawn from this topic with some probability $p(w_n|z)$. Both probabilities are modeled by multinomial distributions. A goal of pLSA is to compute a joint probability of a particular document and a particular word seen together, which is expressed in Equation 4.3.

$$p(d, w_n) = p(d) \sum p(w_n|z)p(z|d) \quad [\text{BNJ03}] \quad (4.3)$$

This approach has several disadvantages. One problem arises by using the parameter d , which is nothing but a dummy index in a list of documents. It results in a fact that the model only learns $p(z|d)$ for documents in the training set and is useless with new documents, which

are used in the test set. Another problem with pLSA is a linear growth of parameters. The larger the dataset, the more parameters are needed. This can result in overfitting.

4.2.6. LDA: Latent Dirichlet Allocation

Latent Dirichlet Allocation, or *LDA*, is a generative Bayesian model, which was introduced by David M. Blei, Andrew Y. Ng, and Michael I. Jordan in 2003 [BNJ03]. This model extends the pLSA model and eliminates the problems described in the previous section. Consider Figure 4.6 with the following notation:

- M is a total number of documents,
- N is a total number of words in a document,
- w is a single word,
- z is a topic variable,
- θ is a topic distribution per document,
- α is a Dirichlet prior on a per-document topic distribution level
- β is a Dirichlet prior on a per-topic word distribution level

This model has three layers indicated by rectangles. A smaller rectangle represents a word layer. A larger rectangle indicates a document level. The space outside the large rectangle represents a corpus level. These rectangular boxes define, which parameter have to be applied on which level.

The parameter w is in a grey circle, which means that this is the only observable parameter. In comparison to pLSA, LDA has a new parameter θ , which indicates a topic distribution. This parameter is Dirichlet distributed. Another new parameters are α and β . A high α indicates that each document is likely to contain the mixture of the most of the topics. A low α , in its turn, denotes that a document contains only a few topics. Similar to α , a high β indicates that each topic contains the mixture of the most of the words, while a low β indicates that a topic contains a mixture of a few words.

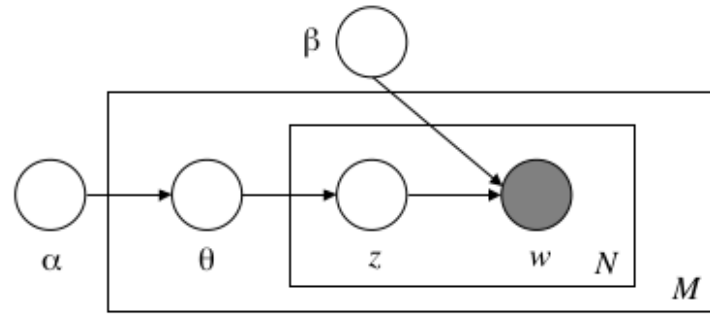


Figure 4.6.: A plate notation of an LDA model [BNJ03].

LDA first randomly assigns each word in each document to one of the topics. Then for each document d , it assumes that all topic assignments except for the current one are correct. The model calculates two proportions: a proportion of words in document d that are currently assigned to topic $z = p(z|d)$, and a proportion of assignments to topic z over all documents that come from this word $w = p(w|z)$. In next step, these two proportions are multiplied, and the model assigns to word w a new topic based on that probability $p(z|d)p(w|z)$. After this step, the assignment of the current word is updated and the whole process is repeated until a steady state, in which assignments make sense, is reached.

4.2.7. NMF: Non-negative Matrix Factorization

This model is an implementation of the approach explained in Subsection 4.2.1. For a better recollection, go back to Figure 4.3. NMF represents a dataset as a *Term x Document* matrix X and decomposes it into a product of two smaller matrices W and H . NMF assumes that all entries within these matrices are not negative. In order to decompose the matrix, the following minimization problem is to be solved:

$$\text{minimize } ||X - WH||^2, \text{ w.t. } W, H, \text{ s.t. } W, H > 0 \text{ [LS01]} \quad (4.4)$$

However, this problem can not be solved analytically. Instead, the decomposition is approximated numerically to find a local minimum. To approximate $X \approx WH$, Lee and Seung's multiplicative update algorithm [LS01] is used, which is based on gradient decent. First,

the matrices W and H are randomly initialized. In each iteration, one matrix is fixed and the other one is updated. Thus, if the matrix W was fixed, the matrix H is updated, and vice versa. This approach leads to a fast convergence of the model.

NMF has proved to be an efficient model with an advantage of a small amount of parameters which have to be tuned.

4.2.8. Topic Modeling Tools

A huge amount of data is being generated every hour. A necessity to obtain the most significant and relevant information triggered a rapid development of different tools for topic modeling. These tools enable users to retrieve hidden topics at the click of a button.

One of the examples of such tools is *Gavagai* [Gav]. This tool is fully customizable and offers an automated topic extraction powered by language understanding. Another alternative is the *NaturalText A.I.* [Nat]. It uses unsupervised learning and symbolic, non-statistical A.I. to extract hidden information from long-form text data. One more example of topic modeling tools is *Semantria* [Sem]. It is a natural language processing API that enables theme analysis, summarization, sentiment analysis, categorization and other services.

Though these solutions deliver impressive results, they are developed for large businesses which means that they are expensive and created to deal with a large amount of text data. This makes them to be inapplicable in the context of online discussions and leads to a decision to develop a custom tool that can meet the needs of *D-BAS* argumentation system.

4.3. Automatic Text Summarization

Text summarization is a challenging task, since it is unclear how to select the most relevant information from the original text. Furthermore, it is unclear how to express this crucial information in the summary itself.

In general, text summarization approaches can be divided into three main branches. Figure 4.7 shows a tree representation of this division.

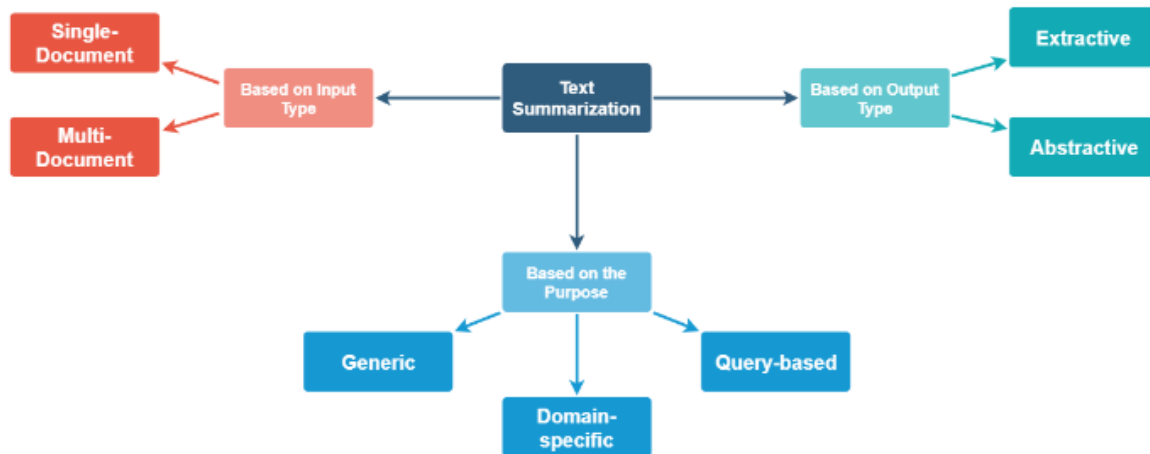


Figure 4.7.: Types of text summarization approaches. Graphic taken from: [Cha18].

Depending on the input type, text summarization can be performed either on a single document or on multiple documents.

Based on the purpose, text summarization can be *domain-specific*, *generic* or *query-based*. Domain-specific text summarization requires domain-specific knowledge for a model to produce an accurate output text. Generic text summarization is a general way to produce a summary without making any assumptions about the field the original text comes from. The goal of a query-based text summarization is to find relevant information, which answers queries from the input text.

Depending on the output type, text summarization can be divided into *extractive* or *abstractive* [All+17]. Abstractive text summarization imitates a human-like summary, using new words, phrases or/and sentences. Abstractive method implies that the model understands the context of the input data and creates new sentences to generate the output text. This approach is the most challenging one, since it requires a vast amount of domain knowledge.

Extractive text summarization delivers the resulting text by means of extracting the most representative sentences from the input text and combining them together. With this approach, the model calculates the similarity between a pair of sentences, weights them and picks the sentences with the highest weightings for the output text. Though, this approach faces such problems as lack of balance and lack of coherence, it is still the mostly widely used and efficient one.

Depending on the type of approach, text summarization techniques can be divided into several groups. Some of them are explained in the subsequent subsections.

4.3.1. Position-based Approach

The *Position-based* approach assumes that the most relevant sentences appear in predefined positions. For instance, after analyzing 200 paragraphs, Baxendale [Bax58] noticed in his experiment that the first and the last sentences of a paragraph are relevant. This experiment is considered to be the first one in the field of the automatic text summarization.

Other studies showed, that in scientific papers, the most relevant sentences occur in the introduction and in the conclusion, while in the newspapers the first sentences have the highest relevance.

4.3.2. Term-based Approaches

The *Term Frequency (TF)* approach belongs to unsupervised extractive algorithms. There are different ways how to compute the term frequency.

The *Raw term frequency* method for text summarization corresponds to the TF approach for topic modeling from Subsection 4.2.3. One of the first works in the field of the term-based text summarization was Luhn's [Luh58] paper, in which he came up with a more advanced technique. This technique used raw term frequency, but assumed that the terms, which appeared the most, and the terms, which appeared the least within a text, were not relevant for further calculations. According to his research, only the terms with a middle-ranged frequency were to be analyzed.

Boolean frequency assigns the weight of the term i as $w_i = 1$, if the term is in the document, and $w_i = 0$, if the term is not found in the document, as shown in Equation 4.5.

$$w_i = \begin{cases} 1 & , \text{ if term } i \text{ is in the document} \\ 0 & , \text{ otherwise} \end{cases} \quad (4.5)$$

Another way to compute the term frequency is to use *log frequency*, which lessens the importance of the term with high frequency.

$$w_i = \begin{cases} 1 + \log(tf_i) & , \text{ if } tf_i > 0 \\ 0 & , \text{ otherwise} \end{cases} \quad (4.6)$$

Augmented frequency, presented in Equation 4.7, reduces the advantage of long documents over short ones. The factor k is set to 0.5 for short documents [WGG17].

$$w_i = k + (1 - k) \frac{tf_i}{\max_i(tf_i)} \quad (4.7)$$

Another term-based technique for the text summarization is *Term Frequency - Inverse Document Frequency (TF-IDF)* approach, which corresponds to the TF-IDF approach for topic extraction from Subsection 4.2.3.

4.3.3. Graph-based Approach: TextRank

Term-based methods are simple to implement but in most cases fail to capture the semantic relations between terms. *Graph-based* approach represents each sentence as a vector and computes their connectivity using the sentence similarity, producing a graph with each sentence as a node. One of the advantages of graph-based approaches is that these models can be used to summarize the texts independently of their language.

One of the most performative graph-based algorithms for text summarization is *TextRank*

[MT04]. In the core of this method lies the well-known Google's *PageRank* algorithm, which was primarily developed for measuring the importance of web pages in search results [BP98]. In *PageRank*, web pages are used to represent vertices of a graph, and edges denote the transition probability of a user going from one page to another. *TextRank*, in its turn, replaces web pages through sentences to represent nodes.

TextRank algorithm works as follows. In case of a multiple document text summarization, the documents have to be first combined into a single text. This step is omitted, if the input consists of only a single text. In the next step, this text is split into tokens and a common text preprocessing techniques are carried out. For each sentence a sentence vector representation is constructed. From the representation vectors a similarity matrix is computed. The similarity between two sentences is defined through a function of their content overlap, given in Equation 4.8.

$$\text{Similarity}(S_i, S_j) = \frac{|w_k | w_k \in S_i \ \& \ w_k \in S_j|}{\log(|S_i|) + \log(|S_j|)} \quad \text{[MT04]} \quad (4.8)$$

The overlap between two sentences S_i and S_j is a number of common terms w_k in both sentences, which is normalized by the length of both sentences. This normalization step prevents the biasing of long sentences. There are also other approaches on how to measure the sentence similarity. For instance, some of the *TextRank* implementations use a cosine similarity technique.

After the similarity matrix is computed, the *PageRank* algorithm for scoring is applied to the sentences. To visualize this graph-based approach, consider Figure 4.8 taken from the original paper.

The left part of the image depicts the sentences, on which the algorithm was run. The right side of the image represents the weighted graph after a scoring function has been applied on it. In the last step, the sentences with top scores are taken to produce an output summary text.

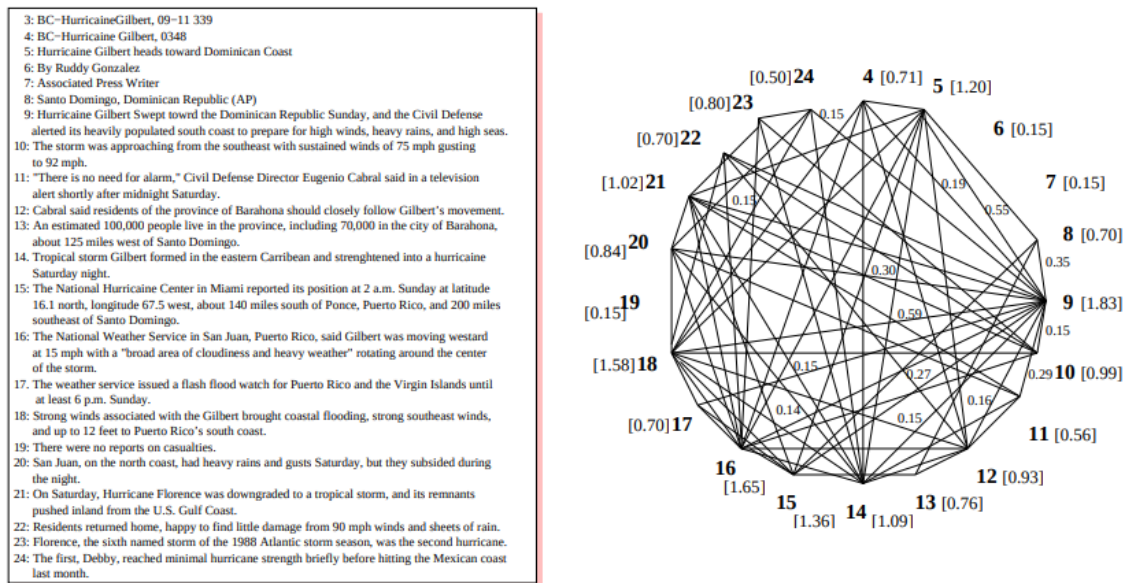


Figure 4.8.: An example text and a weighted graph, obtained from it after TextRank was applied. The original image was taken from [MT04].

4.3.4. Machine Learning Approach

Potentially machine learning approaches can overcome such problems as grammar inconsistency, imbalance of the summary, and the lack of coherence, they still remain domain-dependent and can for this reason be applied on limited tasks. Automatic text summarization can be done using machine learning algorithms. For example, the deep learning *Sequence-to-Sequence* model can be applied to generate a text summary [SVL14]. Recently OpenAI, an artificial intelligence research laboratory, released *GPT-3* (Generative Pre-trained Transformer 3) [Gpt], an API which allows accessing to new models to produce a human-like text. This API can be used for such tasks as text generation, language improvement, Q&A, automatic translation, keywords extraction.

4.4. Automatic Keywords Extraction

There are numerous approaches for keyword extraction. The most widely used methods are unsupervised since they don't require any domain knowledge. The most part of the unsupervised keyword extraction methods have at their core the same techniques, which are

used for the text summarization and which were described in the previous section.

4.4.1. Term-based Approaches

Term-based approaches, such as TF and TF-IDF, make use of the term frequencies in a way, which was already described in Subsection 4.3.2. These approaches have the same advantages and disadvantages as in the case of the text summarization: they are language-independent and do not require additional domain knowledge, but they fail to capture such crucial aspects as structure, relations between terms, meaning, etc.

4.4.2. Term-based Approaches: RAKE

Another statistical algorithm for keywords extraction is called *RAKE*, or *Rapid Automatic Keyword Extraction*. This algorithm makes use of word collocations (adjacent terms, which usually go together) and co-occurrences. RAKE receives several parameters as an input: a stopword list, a set of phrase delimiters, and a set of word delimiters [Ros+10].

Consider an example text in Figure 4.9. In the first step, RAKE uses the word delimiters to split the input text into an array of terms.

Compatibility of systems of linear constraints over the set of natural numbers

Criteria of compatibility of a system of linear Diophantine equations, strict inequations, and nonstrict inequations are considered. Upper bounds for components of a minimal set of solutions and algorithms of construction of minimal generating sets of solutions for all types of systems are given. These criteria and the corresponding algorithms for constructing a minimal supporting set of solutions can be used in solving all the considered types of systems and systems of mixed types.

Figure 4.9.: An example text for keywords extraction. The original image was taken from [Ros+10].

In the next step, these terms are split into words based on the phrase delimiters and the stopwords, provided as the input parameter, are removed, and only candidate expressions remain. These candidates are shown in Figure 4.10.

Then, using the obtained candidate keywords, RAKE creates a co-occurrence matrix similar

Compatibility – systems – linear constraints – set – natural numbers – Criteria – compatibility – system – linear Diophantine equations – strict inequations – nonstrict inequations – Upper bounds – components – minimal set – solutions – algorithms – minimal generating sets – solutions – systems – criteria – corresponding algorithms – constructing – minimal supporting set – solving – systems – systems

Figure 4.10.: Keywords candidates. The original image was taken from [Ros+10].

to an example matrix shown in Figure 4.11. Every row contains the number of times, each term co-occurs with other terms in the candidate expressions.

	algorithms	bounds	compatibility	components	constraints	constructing	corresponding	criteria	diophantine	equations	generating	inequations	linear	minimal	natural	nonstrict	numbers	set	sets	solving	strict	supporting	system	systems	upper
algorithms	2						1																		
bounds	1																								1
compatibility		1																							
components			2																						
constraints				1									1												
constructing					1																				
corresponding	1					1																			
criteria								2																	
diophantine									1	1			1												
equations									1	1			1												
generating											1		1						1						
inequations												2				1					1				
linear					1				1	1		2													
minimal											1		3					2	1					1	
natural															1	1									
nonstrict											1				1	1									
numbers															1	1									
set													2				3							1	
sets											1		1					1							
solving																			1						
strict											1									1					
supporting														1								1			
system																							1		
systems																								4	
upper	1																								1

Figure 4.11.: An example co-occurrence matrix, which was built using the cleaned terms. The original image was taken from [Ros+10].

The co-occurrence matrix is then used to score the terms. The score of a term w can be computed using Equation 4.9.

$$score(w) = \frac{deg(w)}{freq(w)} = \frac{\text{number of times } w \text{ co-occurs with other terms}}{\text{number of times } w \text{ occurs in candidate expressions}} \quad (4.9)$$

In the next step, the candidate expressions are scored by calculating the sum over the scores of the individual terms. The results of this step for the candidate keywords from Figure 4.10 are shown in Figure 4.12.

	algorithms	bounds	compatibility	components	constraints	constructing	corresponding	criteria	diophantine	equations	generating	inequations	linear	minimal	natural	nonstrict	numbers	set	sets	solving	strict	supporting	system	systems	upper
deg(w)	3	2	2	1	2	1	2	2	3	3	3	4	5	8	2	2	2	6	3	1	2	3	1	4	2
freq(w)	2	1	2	1	1	1	1	2	1	1	1	2	2	3	1	1	1	3	1	1	1	1	1	4	1
deg(w) / freq(w)	1.5	2	1	1	2	1	2	1	3	3	3	2	2.5	2.7	2	2	2	2	3	1	2	3	1	1	2

Figure 4.12.: Scores for candidate expressions from Figure. The original image was taken from [Ros+10].

In the last step, a required number of candidates with the top scores is returned.

4.4.3. Graph-based Approach: TextRank

Graph-based techniques for keywords extraction create a graph from the input text, in which vertices correspond to single terms. One of the graph-based algorithms used for keywords extraction is TextRank, which was described in Subsection 4.3.3. During the preprocessing stage, TextRank applies Part-of-Speech filters to tag each term. Further unnecessary terms like, for example, prepositions can be removed. The preprocessed tokens are then used to build a unidirectional unweighted graph, in which edges are drawn between terms which co-occur within a predefined window size. After this step, the PageRank ranking algorithm for unweighted graphs is applied on the graph. Finally, the top-ranked words are chosen to generate the output keywords [MT04].

As in the case of the text summarization, graph-based approaches have an advantage of delivering a good performance without the necessity to take the language into account.

4.4.4. Machine Learning Approach

As well as in the case of the automatic text summarization, keywords extraction can be done by using machine learning techniques, such as Support Vector Machine or algorithms from a deep learning field. For instance, Wu et al. proposed a least square support vector machines (LS-SVM) model, which extracts keywords from scientific literature [Wu+07]. Also, the Sequence-to-Sequence model, which was already mentioned in Subsection 4.3.4, can be used for automatic information extraction [SVL14]. Models provided by the *GPT-3* API and

mentioned in Subsection 4.3.4 can be used to generate keywords.

Though these approaches have a huge potential, their main disadvantage is their domain-based nature, which makes models be applicable only for very specific tasks.

Chapter 5.

Implementation

In the previous chapter, the main approaches and algorithms used in the fields of topic modeling, text summarization, and keywords extraction, were introduced. This chapter gives an overview of the main decision-making processes behind the implementation of the proposed tool.

5.1. General Project Structure

Since *Django Framework* already provides a logical and loosely coupled architecture by encapsulating every unit within an application (*app*), the project has the following partially predefined structure which is shown in Figure 5.1. Every unit is responsible for one particular task and, for this reason, modifications within one app will only have a small effect on other apps.

This project consists of five main apps and several additional modules with auxiliary functions. *Topicextractor* app includes general *Django* settings and a definition of basic routes. *Keywords* app is responsible for generation and visualization of keywords. *Summary* app manages the extraction of relevant sentences for a summary and its visualization. *Topics* app generates topics from a dataset and visualizes them. Each of these three apps contains

one additional *api*-app, which allows an interaction between this service and external software components. *Pages* app, in its turn, has an auxiliary function and is responsible for a representation of a home page from which a user can navigate to one of the three modules mentioned above. Figure 5.2 displays the home page of the application.

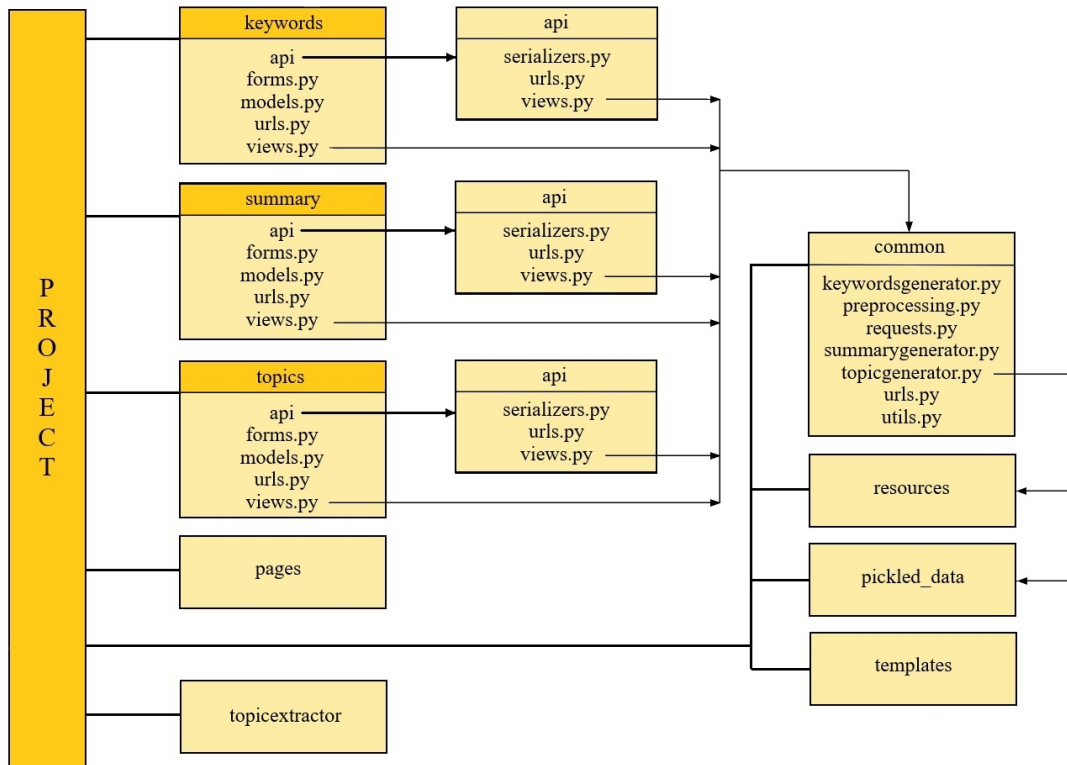


Figure 5.1.: A general project structure.

All apps have a similar structure. Every app has a model which can be defined within a *model.py*-file. This model serves as an intermediary between a database and a user interface. The model defines in which form the data has to be stored in the database.


The *views.py*-files are responsible for the user interface of the applications by rendering *HTML*-templates, which control user interactions and are gathered within an auxiliary *templates* module.

Additionally, the *forms.py*-files make it possible to define an easy mechanism for collecting the parameter input by providing different built-in widgets, which are rendered within templates by *Django* views. These parameters can later be used for generation of summaries, topics and keywords.

Home Topics Summary Keywords API

Welcome to Topic Extractor and Summarizer!


From the list below you can choose the section you are interested in.



Topic Extractor

Learn what participants are talking about in D-BAS discussions. With this topic generator you can extract the most significant words grouped into topics.


[Go to Topic Generator](#)



Summarizer

Don't want to read the whole discussion to learn what it is about? There is a solution! Use a summarizer to extract the summarized version of the debate.

[Go to Summarizer](#)



Keywords Extraction

Extract the keywords from the D-BAS discussions.

[Go to Keywords Generator](#)

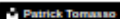

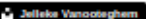
All images belong to:   

Figure 5.2.: The home page of the application.

Subsequently, the routing within the entire project is enabled by mapping the url patterns to the views, which can be defined within *urls.py*-files.

All *api*-apps resemble in structure the above described apps with one difference; instead of having models connected to the database, they have *serializers* which convert *Django* models into other formats for the data to be sent over an *API*. In this case, it is the *json*-format. Serializers can be defined within *serializers.py*-files.

A logic behind text preprocessing and further summary, topics and keywords generation is encapsulated within the *common*-module. Both *resources* and *pickled_data* modules are used by this module to enable the usage of pre-trained models for text classification. The *common*-module will be introduced in more detail in the subsequent section.

In order to make the routing within the project more transparent, the routes were split into two parts. The main routes defined in the *urls.py*-file of the *topicextractor* module and the routes, which are specific for every single app. Each app was given a unique name to avoid possible path collisions between modules.

5.2. Topic Modeling: *topics* app

One of the main problems in topic modeling is the size of a dataset. Since the size of a discussion can not be predicted in advance, a decision was made to enhance the proposed tool with the ability to both train a model on the discussion statements and to use the already pre-trained model and perform a text classification by treating the discussion statements as unseen documents. If the dataset is big enough, a training of the model on this data will produce well-defined topics. If the dataset is small, a pre-trained model may help in finding appropriate topics.

Another challenge behind topic modeling is the choice of packages, models and their parameters. Depending on the dataset, the same model provided by the same package will deliver results which differ in their quality. Since the proposed tool has to be universally applicable, a decision was taken to enable users to define several parameters which give them freedom to fine-tune the models. In such a way users can decide which results are the most suitable for their tasks.

The *topics* app is responsible for generating topics. It provides a graphical interface that enables users for input parameters to retrieve topics from a discussion. Figure 5.3 illustrates the graphical interface for the input parameters.

Figure 5.3.: Graphical interface of the *topics* module.

5.2.1. Parameters

There are seven parameters, which the user has to provide in order to perform topic modeling: *discussionID*, *pretrainedModel*, *package*, *inputModel*, *nGrams*, *numTopics*, and *user*. These parameters can be passed with the following path: */topics/*.

- *discussionID*: This parameter is required to make a further request to the *D-BAS API* to get the discussion statements, which will further be used either as a dataset to train a model or as unseen documents, which will be fitted into the pre-trained model.
- *pretrainedModel*: This parameter defines, if a pre-trained model has to be used for the topic generation or not. By setting it to *false*, the model will be trained only on the discussion data. Otherwise, the discussion statements will be classified according to the pre-trained model.
- *package*: Users can also define which package and model have to be used to perform the topic generation. This parameter can be set either to use the *sklearn* or the *gensim* packages. By default, this parameter is set to *sklearn*.
- *inputModel*: Depending on the choice of the package, the user can set this parameter to be *NMF*, *LDA* or *LSA*. These models were chosen, because they already proved to

deliver reasonable results on the datasets of different sizes. By default, the value of this parameter is set to *LDA*, since this model is provided by both *sklearn* and *gensim* packages.

- *numTopics*: Through this parameter, the user can control the number of generated topics. The value of this parameter has to be strictly positive. If no value is provided, the proposed tool will by default set this parameter to 3. Since the discussions are more likely to be short, a larger number of topics can only lead to poor results.
- *nGrams*: By setting this parameter to *true*, the user enable the usage of bi-grams by training a model. Otherwise, one-grams will be used.
- *user*: Since there is no user authentication mechanism, which could enable filtering the data within the database, this parameter is required. After the topics have been generated, they are stored in the database, where one of the fields is set to the value provided by this parameter.

5.2.2. Use Cases

Since all parameters have been introduced, consider the following use case. Figure 5.4 displays a piece of the discussion with the *discussionID* = 4. This discussion will be used to generate topics.

```
- we should introduce an admission restriction
- the demand for the course is too large, so that a restriction must be introduced
- many students register themselves without having the necessary skills
- a comparability of the high school grades is not given
- the capacity of the university should be increased instead of excluding new students
- the course standards should be increased
- the students will receive a professionally higher degree
- the number of students can be reduced in a natural way
- many students are already dropping out
- drop-out rate should not be lowered, but the number of less talented freshmen should be lowered
- previous graduates still have deficits in many areas
- students, who do not drop out, have too little knowledge to start their professional life yet
- the level of requirement is not printed on the certificates
- the secondary school certificate does not contain a lot of information regarding computer science skills, since only a few schools teach computer science in a serious fashion
- I do have the impression (without being able to give specific numbers), that students discontinue their studies more often because of problems with mathematics rather than with computer science
- we do not have a sufficient amount of professorships in computer science to offer these additional courses
- with the increased number of students we should be able to let computer science professors hold those lectures
```

Figure 5.4.: A piece of the discussion (*discussionID* = 4) used to generate topics.

As seen in Figure 5.5, four topics will be generated for the discussion from Figure 5.4 with

discussionID = 4. For this, the *sklearn* implementation of the *NMF* model will be used. The topics were generated without bigrams and without any pre-trained model. The generated topics are displayed below the parameter bar on the same graphic.

127.0.0.1:8000/topics/search/?user=test_user&discussionID=4&pretrainedModel=false&package=sklearn&inputModel=NMF&nGrams=false&numTopics=4

Home Topics Summary Keywords API ▾

Username: test_user Discussion ID: 4 Pretrained model: False ▾ Package: sklearn ▾

Topic Model: NMF ▾ N-grams: False ▾ # Topics: 4

Get Data From GraphQL

Topic
study, school, secondary, degree, good, start, test, mark, master, suitability, determine, sample, visit, science, compulsory

Topic
lecture, script, mathematic, online, question, offer, subject, content, slide, note, year, visit, exercise, learning, time

Topic
student, exercise, require, number, test, question, task, solution, increase, possible, group, capacity, time, pass, discussion

Topic
computer, science, mathematic, course, subject, scientist, important, close, offer, case, secondary, proof, refer, point, school

Figure 5.5.: The parameter bar and four generated topics for the English discussion (*discussionID* = 4).

If one of the parameters is set to a wrong value — for example, a wrong format — the proposed topic extractor will generate an error. Consider Figure 5.6 which illustrates a case when the user changes the url and sets the number of topics to be a negative number. This part of the url is underlined in red. The proposed tool will also generate similar error messages if a user provides invalid values for other parameters or if a response from the *D-BAS* was empty.

127.0.0.1:8000/topics/search/?user=test_user&discussionID=4&pretrainedModel=false&package=sklearn&inputModel=NMF&nGrams=false&numTopics=-3

Home Topics Summary Keywords API ▾

Username: Discussion ID: Pretrained model: Package:

Topic Model: N-grams: # Topics:

Error

Number of topics has to be a strictly positive integer but not larger than 50.

Figure 5.6.: Example of an error case when a user provides an invalid value for the *numTopics* parameter.

5.2.3. Data preprocessing

In the preprocessing step, the techniques described in Section 4.1 were used. Stopwords, punctuation, special characters and words consisting of less than three characters were removed and the remaining data was lemmatized. To clean the data, the *spacy* package was used, because it provides trained language models for both English and German languages, which makes a part-of-speech tagging and subsequent data cleaning easier. The *spacy* stopwords list was extended by the stopwords provided by the *nlTK* package. Custom stopwords were also added. After the input text was cleaned, the tokenized data is vectorized to be later used to train a model.

5.2.4. Pre-trained Models: External Datasets

At the moment, *D-BAS* has discussions in two languages: mostly in German but also in English. For this reason, two different datasets had to be used to pre-train two different models. *D-BAS* discussions are narrowly specialized while preserving a colloquial way of speech in scientific topics. For this reason, an attempt was made to create datasets, which could reflect these aspects.

For the English dataset one of the Wikipedia dumps from 01.08.2020 [Enw] was used. To

build a German dataset, a random Wikipedia dump [Dew] was used. This data was merged with *Ten Thousand German News Articles Dataset* [10k], which comprises 10273 German language news articles. The merged corpus was then preprocessed by using techniques described in Section 4.1.

Wikipedia dumps were chosen to reflect the scientific, narrowly specialized side of *D-BAS* discussions. These dumps also include the comments of their contributors, which in its turn has to reflect a conversational style of discussions. Both dumps were chosen randomly to reflect the fact that a topic of a discussion can not be predicted in advance. The *Ten Thousand German News Articles Dataset* was used to capture a not chronological and abrupt style, which is usually used in news articles and can be found in discussion statements.

After the preprocessing step described in Subsection 5.2.3, the data was then used to train the models from both packages *sklearn* and *gensim* with different sets of parameters to find the optimal ones.

Table 5.1 illustrates the set of parameters that were used to train the models from the *sklearn* package. The topics from pre-trained models were then assessed by a human evaluation.

Model	NMF	LDA	LSI
Iterations	300, 500, 750, 1000	300, 500, 750, 1000	10, 30, 100, 200, 300
Beta loss	Frobenius, Kullback-Leibler	-	-
Num. topics	20, 40, 50, 60	20, 40, 50, 60	20, 40, 50, 60
Alpha	0.1, 0.01	-	-
Decay	-	0.5, 0.7, 0.9	-

Table 5.1.: Set of training parameters (*sklearn*).

For the English and German *NMF* models the parameters were set with 500 iterations, 50 topics, “*Frobenius*“ beta loss and alpha 0.1 were chosen. For the English and German *LDA* models the models with 100 iterations, 50 topics and decay 0.7 were taken. The models with 300 iterations and 40 topics were selected for the English and German *LSI*.

All topics of the pre-trained *sklearn LDA* models are provided in Appendices B and D.

Table 5.2 gives an overview of the set of parameters used to train the models from the *gensim* package.

After the evaluation of the produced topics, the following models were chosen. For both

Model	LDA	LSI
Iterations	300, 500, 700, 900	2, 4, 8, 10, 50, 100, 250, 500
Num. topics	20, 40, 50, 60	20, 40, 50, 60
Passes	40, 50, 60, 70	-
Extra samples	-	100, 200, 300

Table 5.2.: Set of training parameters (*gensim*).

English and German *LDA* models, the models with 900 iterations, 40 passes and 50 topics were chosen. For the English *LSI* model the model with 50 iterations, 40 topics and 300 samples was selected. For the German *LSI* model the model with 10 iterations, 30 topics and 300 samples was taken.

Examples of the pre-trained *gensim LDA* topics are provided in Appendices A and C.

In case of the *pretrainedModel* parameter is set to *true*, a text classification on the *D-BAS* data is performed. The original approach was to use the whole data as a single document to fit into the pre-trained model. Since the topics from the external dataset only reflect general topics and fail to capture fine-grained information of the discussions, this approach failed to assign the discussion to the correct pre-trained topics, which will be shown in Chapter 6.

Under this consideration, an approach to treat each statement as a new document was applied. In order to identify the most significant topics, the topics scores are tracked within a dictionary. After all statements were classified, the topics with the highest scores were retrieved and stored within a database.

5.2.5. Topic Model

After the topics were generated, the *Topic* model is used to store the data in the database. This model is defined in the *models.py*-file of the *topics* module. The model has the following fields:

- *user*: This field stores a username provided by the user.
- *content*: This field stores a generated topic as a string.

- *error*: This field stores the information about possible errors which can occur during the topic generation process. For instance, if the user provides an invalid package name, this field will be set to “*Invalid package name*“ error. By default, this field is set to an empty string.
- *timestamp*: This field keeps track of a time stamp of the generated topic.
- *id*: This field is generated automatically by the database.

5.2.6. API

Each serializer of this *API* has the same fields as the *Topic* model from the previous section. The *API* of this module has two related endpoints, each of which use *GET* request methods to retrieve resources.

- */api/topics/*: By providing the same parameters as in Subsection 5.2.1 to this path, an external service can retrieve a list of generated topics with the amount of topics equal to the *numTopics* parameter.
- */api/topics/{pk}*: By providing a primary key parameter to this endpoint, a topic with the specified *primary key* can be retrieved.

Figure 5.7 illustrates a request to the first endpoint. In this example, the user requests a resource generated by the *NMF* model of the *sklearn* package, which consists of three topics. These topics will be retrieved from a discussion with the id number 4. No pre-trained model will be used for topic generation.

Similarly, Figure 5.8 demonstrates how a topic with a particular *primary key* can be retrieved. In this example, a resource with the *primary key* = 24 is requested.

```

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": [
    {
      "user": "test_user",
      "id": 24,
      "content": "lecture, script, mathematic, question, online, offer, content, exercise, slide, note, subject, year, visit, time, learning",
      "error": "",
      "timestamp": "2020-09-05T17:59:22.987986Z"
    },
    {
      "user": "test_user",
      "id": 23,
      "content": "student, study, exercise, test, require, start, number, good, task, work, group, visit, increase, beginning, question",
      "error": "",
      "timestamp": "2020-09-05T17:59:22.982217Z"
    },
    {
      "user": "test_user",
      "id": 22,
      "content": "computer, science, mathematic, course, subject, study, scientist, school, secondary, test, important, close, able, case, offer",
      "error": "",
      "timestamp": "2020-09-05T17:59:22.932425Z"
    }
  ]
}

```

Figure 5.7.: Topics List API with the following parameters: *user* = test_user, *discussionID* = 4, *trainedModel* = false, *package* = sklearn, *inputModel* = NMF, *nGrams* = false, *numTopics* = 3

```

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": [
    {
      "user": "test_user",
      "id": 24,
      "content": "lecture, script, mathematic, question, online, offer, content, exercise, slide, note, subject, year, visit, time, learning",
      "error": "",
      "timestamp": "2020-09-05T17:59:22.987986Z"
    }
  ]
}

```

Figure 5.8.: Topics Retrieve API requesting a resource with the *primary key* = 24.

5.3. Text Summarization: *summary* app

The *summary* app handles the logic behind the summary generation process. On Figure 5.9, an example of the graphical interface for the input parameters for summarization is given.

The figure shows a web form for the 'summary' module. It consists of five input fields arranged in two rows. The first row contains 'User' (text input), 'Statement ID' (text input), and 'Choose Model' (dropdown menu). The second row contains 'Consider Structure' (dropdown menu) and 'Set Ratio' (dropdown menu). Below the form is a blue button labeled 'Get Data From GraphQL'.

Figure 5.9.: Graphical interface of the *summary* module.

5.3.1. Parameters

The following five parameters have to be provided in order to summarize the input data: *positionID*, *inputModel*, *considerStructure*, *ratio*, and *user*.

- *positionID*: In comparison to the topic extractor discussed in Section 5.2, which works on the discussion level, the proposed summarizer works on the level of single positions. This allows the user to retrieve summaries both for the complete discussion and for its subtrees. For this reason, this parameter has to be provided.
- *inputModel*: The proposed tool enables users to define, which of the algorithms has to be used to produce a summary by setting this parameter to one of the following options: *gensim*, *textrank*, *TF* or *TF-IDF*. By default, this parameter is set to *gensim*. This option utilizes the *TextRank* algorithm described in Subsection 4.3.3 provided by the *gensim* package. The *textrank* option utilizes the same algorithm provided by the *spacy* module. *TF* and *TF-IDF* options utilize the approaches described in Subsection 4.3.2. Their implementation is explained in the subsequent section.
- *considerStructure*: If this parameter is set to *true*, the structure of the position tree will be taken into account while producing the summary.
- *ratio*: This parameter defines which amount of text the summary will consist of. By default, this parameter is set to 0.5 making sure that the summary will consist of a half of the original data.

- *user*: As in case of the topic extractor, this parameter is required to map the produced summary to the user within the database.

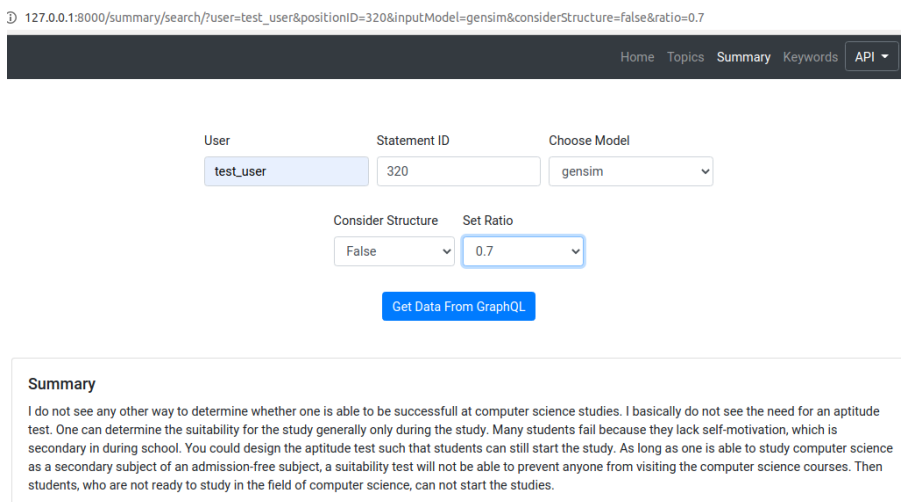
5.3.2. Use Cases

In order to get a clearer understanding of how the proposed summarization tool can be used, consider the following use cases. Figure 5.10 displays the statements from the position that will be used to demonstrate different use cases.

1. I do not see any other way to determine whether one is able to be successful at computer science studies
2. studying not only imparts knowledge, but also key competencies, which may be important in other disciplines
3. one can determine the suitability for the study generally only during the study. Many students fail because they lack self-motivation, which is secondary in during school
4. it would be helpful for students to get an assessment at the beginning. You could design the aptitude test such that students can still start the study
5. this will be shown in the first semesters anyway
6. as long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses
7. then students, who are not ready to study in the field of computer science, can not start the studies
8. I basically do not see the need for an aptitude test
9. one takes valuable time from the students by this

Figure 5.10.: The position (*positionID* = 320) from the *D-BAS* discussion in English which was later used to extract summaries for the evaluation step.

Consider the following parameters: *positionID* = 320, *inputModel* = gensim, *ratio* = 0.7, *considerStructure* = false. By providing these parameters to the graphical interface from Figure 5.9, the result shown in Figure 5.11 can be obtained.



127.0.0.1:8000/summary/search/?user=test_user&positionID=320&inputModel=gensim&considerStructure=false&ratio=0.7

Home Topics Summary Keywords API

User: test_user Statement ID: 320 Choose Model: gensim

Consider Structure: False Set Ratio: 0.7

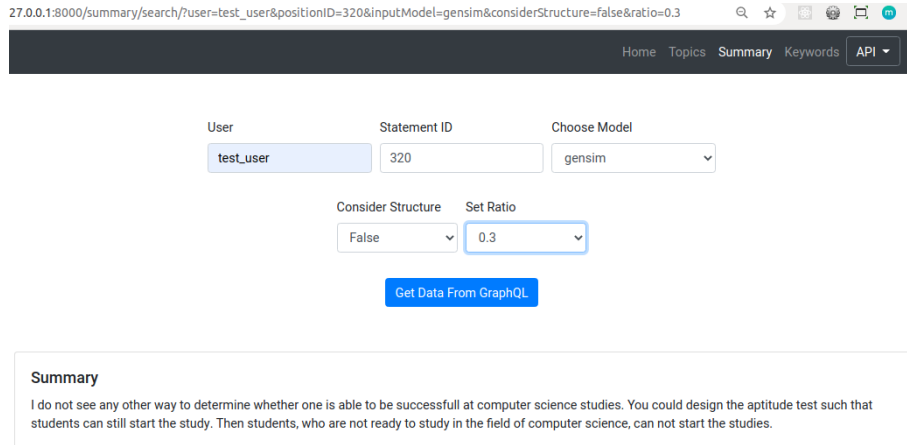
Get Data From GraphQL

Summary

I do not see any other way to determine whether one is able to be successful at computer science studies. I basically do not see the need for an aptitude test. One can determine the suitability for the study generally only during the study. Many students fail because they lack self-motivation, which is secondary in during school. You could design the aptitude test such that students can still start the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. Then students, who are not ready to study in the field of computer science, can not start the studies.

Figure 5.11.: Summary generated with the following parameters: *positionID* = 320, *inputModel* = gensim, *ratio* = 0.7, *considerStructure* = false.

By preserving the same parameters and changing only the *ratio* parameter to be 0.3, a summary of a smaller size can be generated as seen in Figure 5.12.



27.0.0.1:8000/summary/search/?user=test_user&positionID=320&inputModel=gensim&considerStructure=false&ratio=0.3

Home Topics Summary Keywords API

User: test_user, Statement ID: 320, Choose Model: gensim

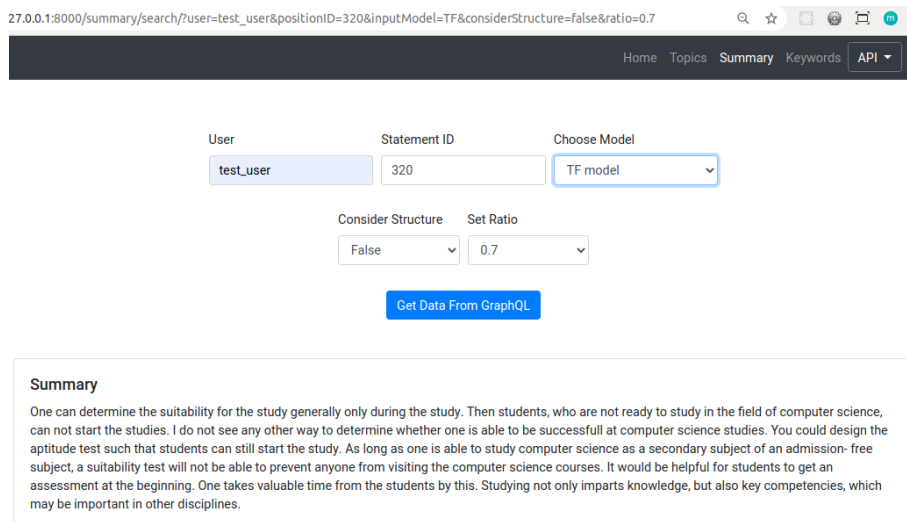
Consider Structure: False, Set Ratio: 0.3

Get Data From GraphQL

Summary
I do not see any other way to determine whether one is able to be successful at computer science studies. You could design the aptitude test such that students can still start the study. Then students, who are not ready to study in the field of computer science, can not start the studies.

Figure 5.12.: Summary generated with the following parameters: *positionID* = 320, *inputModel* = gensim, *ratio* = 0.3, *considerStructure* = false.

If a user takes the parameters from the first example and changes the *inputModel* parameter to *TF*, the user obtains the following results depicted in Figure 5.13. It is obvious that both demonstrated summaries vary greatly depending on the model.



27.0.0.1:8000/summary/search/?user=test_user&positionID=320&inputModel=TF&considerStructure=false&ratio=0.7

Home Topics Summary Keywords API

User: test_user, Statement ID: 320, Choose Model: TF model

Consider Structure: False, Set Ratio: 0.7

Get Data From GraphQL

Summary
One can determine the suitability for the study generally only during the study. Then students, who are not ready to study in the field of computer science, can not start the studies. I do not see any other way to determine whether one is able to be successful at computer science studies. You could design the aptitude test such that students can still start the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. It would be helpful for students to get an assessment at the beginning. One takes valuable time from the students by this. Studying not only imparts knowledge, but also key competencies, which may be important in other disciplines.

Figure 5.13.: Summary generated with the following parameters: *positionID* = 320, *inputModel* = TF, *ratio* = 0.7, *considerStructure* = false.

Similar to the topic generator, the proposed summarizer also generates corresponding errors if invalid parameters are provided. For instance, Figure 5.14 illustrates an error handling if

an invalid *positionID* was provided. In this case, a user changed the url and set this parameter to a negative number. The invalid part of the url is underlined in red.

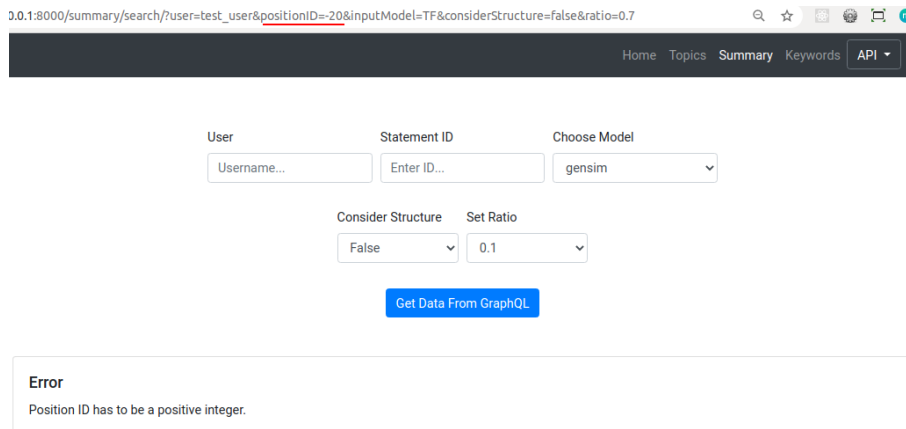


Figure 5.14.: Example of an error case when a user provides an invalid value for the *positionID* parameter.

5.3.3. Input Text Preparation

One of the strengths of *D-BAS* is a graph-like structure of discussions, which stores the information about relations between statements. For this reason, an attempt was made to take this information into account while generating summaries. Consider the following situation depicted on Figure 5.15, in which a position with a recursive tree-like structure and five statements is shown. In this position the statement *A* is a premise for the conclusion *P*. This part of a tree is marked by a green rectangle. But at the same time, the statement *A* is a conclusion for a subtree marked with orange with premises *B*, *C* and *D*.

To capture the above described situation, a decision was taken to assign weights to sentences that are both premises and conclusions while producing the input text. The first approach made in this direction was to look at how many statements are the premises for this particular statement. The more premises this conclusion has, the more attention of the discussion users it invoked. That is why, while producing the input text, such statements were duplicated as many times as many premises it had. For such graph-based algorithm as *TextRank*, the duplication of the sentences will not have any impact on the output summary, because it creates a graph with the sentences and not with single words as nodes. But term-based approaches, such as *TF* and *TF-IDF*, may improve their performance by assigning higher weights to single terms.

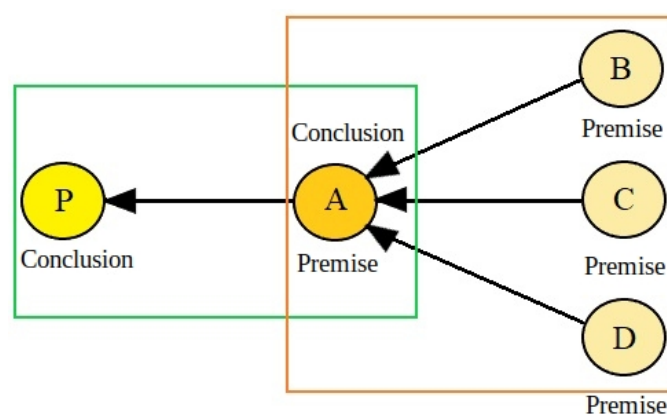


Figure 5.15.: An example of a position structure.

The duplication of the statements caused the situation when short sentences, which do not bear any significant information but have several premises, were biased so much, that they gained the highest scores in comparison to other non-duplicated sentences. Under this consideration, the above described approach was adjusted to double such sentences only once while concatenating them to a single input text.

Additionally, users have an opportunity to generate summaries without taking the position structure into account. This can be done by setting the *considerStructure* parameter to *false*. In this case, algorithms iterate over all statements concatenating them into one text.

5.3.4. Summarization

If the *inputModel* parameter is set to *gensim* or *TextRank* option, the input text is summarized by using the *TextRank* algorithm implemented in either the *gensim* or the *spacy* package. Internally, it creates a similarity matrix of all the sentences from the input text, where the similarity between all the pairs of sentences is stored. After this step, a graph is build with sentences as vertices and a *PageRank* is applied.

If the *inputModel* parameter is set to *TF*, every sentence is preprocessed in the same way as it was already described in Subsection 5.2.3. After this step, a frequency table and a sentence table are created. The frequency table is used to keep track of how often each word within a sentence occurs. The sentence table creates a mapping between the original input sentences and their cleaned and tokenized alternatives. After this step, each sentence is

scored by summing up the number of occurrences of all terms within this sentence. In order to avoid the biasing of long sentences and preventing that they have an advantage over short sentences, the score is divided by the number of all terms from this sentence. Subsequently, the sentences are sorted by their scores and a threshold is defined by multiplying the number of sentences by the provided ratio. This threshold defines how many sentences will be used to generate a summary. In the last step, the sentences with the highest scores are concatenated and the output summary is stored within the *Summary* model, which has the same structure as the *Topic* model from Subsection 5.2.5.

The *TF-IDF* algorithm extends the *TF* approach. After creating the frequency table and the sentence table, a word per sentence table is created. In the *TF-IDF* approach each sentence is treated as a single document. In the next step, an inverse document frequency for each word is computed. For this, a frequency of a term is divided by the number of words in the document. In the next step, a table for documents per words is computed. Next, a term frequency inverse document frequency is calculated and stored within a table. Subsequently, the scores for all sentences are calculated by summing up the scores of all terms within each sentence. These scores are normalized by dividing them by the length of sentences to avoid the biasing of long sentences. In the last step, a threshold is calculated and the sentences with the highest scores are retrieved for the output summary.

5.3.5. API

Serializers of this *API* have the same fields as serializers from *Topic API*. Similarly to the *Topic API*, this module has two related endpoints. In both cases, resources are retrieved by using *GET* request methods.

- */api/summary/*: By providing the same parameters as in Subsection 5.3.1 to this endpoint, users can retrieve a generated summary.
- */api/summary/{pk}*: By providing a primary key parameter to this path, a summary with the specified *primary key* can be retrieved.

For instance, on Figure 5.16 the user makes a request to the first endpoint by requesting a summary of the position with the id = 320. The summary will be generated by the *gensim*

package with the ratio set to 0.5. The structure of the position will not be considered since the corresponding parameter is set to *false*.

```

Summary List Api GET
GET /api/summary/?user=test_user&positionID=320&inputModel=gensim&considerStructure=false&ratio=0.5

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": [
    {
      "user": "test_user",
      "id": 158,
      "content": "I do not see any other way to determine whether one is able to be successfull at computer science studies. You could design the apit",
      "error": "",
      "timestamp": "2020-09-05T20:47:58.296907Z"
    }
  ]
}

```

Figure 5.16.: Summary List API with the following parameters: *user* = test_user, *positionID* = 320, *considerStructure* = false, *inputModel* = gensim, *ratio* = 0.5.

If the user wants to retrieve an already existing summary, it can be done by making a request to the second endpoint as seen on Figure 5.17. In this example, a resource with the id = 160 is requested.

```

Summary Retrieve Api GET
GET /api/summary/160/

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": {
    "user": "test_user",
    "id": 160,
    "content": "as long as one is able to study computer science as a secondary subject of an admission-free subject, a suita",
    "error": "",
    "timestamp": "2020-09-06T08:32:01.440886Z"
  }
}

```

Figure 5.17.: Summary Retrieve API requesting a resource with the *primary key* = 160.

5.4. Keywords Extraction: keywords app

The *keywords* app handles the logic behind the keyword extraction process. It provides a graphical interface for the input parameters, which is displayed on Figure 5.18.

The form consists of five input fields arranged in two rows. The first row contains 'User' (text input), 'Position ID' (text input), and 'Choose Model' (dropdown menu). The second row contains 'Consider Structure' (dropdown menu) and 'Set Ratio' (dropdown menu). A blue button labeled 'Get Data From GraphQL' is centered below the second row.

Figure 5.18.: Graphical interface for the input parameters of the *keywords* module.

5.4.1. Parameters

Users have to provide the following four parameters in order to extract keywords from the input data: *positionID*, *inputModel*, *considerStructure*, *ratio*, and *user*.

- *positionID*: As in case of the summarizer from the previous section, this module works on the level of a single position. This parameter has to be provided in order to perform keywords extraction either on a single position or on the whole discussion.
- *inputModel*: This parameter specifies, which of the algorithms has to be used to extract keywords. This parameter has the following options: *gensim*, *rake* or *custom model*. By default, this parameter is set to *gensim*. The *gensim* option uses the *gensim* package to generate keywords. The *rake* option utilizes the implementation of the *Rake* algorithm described in Subsection 4.4.2. The *custom model* option extracts keywords by using the term-frequency approach from Subsection 4.4.1. Its implementation is explained in the subsequent section.
- *considerStructure*: Equally in case of the text summarization, if this parameter is set to *true*, the structure of the position will be taken into account while extracting the keywords.
- *ratio*: This parameter defines which amount of words will be returned as keywords. By default, this parameter is set to 0.5.
- *user*: This parameter is required to map the produced keywords to the user within the database.

5.4.2. Use Cases

Consider the following use cases in order to get a better understanding of the proposed keyword extractor. For this, the keywords will be generated with different parameters from the same position as in case of the proposed summarizer.

Figure 5.19 shows the keywords which were generated by using the *Rake* algorithm with *ratio* set to 0.3.

The screenshot shows a web browser window with the URL `127.0.0.1:8000/keywords/search/?user=test_user&positionID=320&inputModel=rake&considerStructure=false&ratio=0.3`. The page has a navigation bar with 'Home', 'Topics', 'Summary', 'Keywords', and 'API'. Below the navigation bar, there is a form with the following fields: 'User' (test_user), 'Position ID' (320), 'Choose Model' (Rake), 'Consider Structure' (False), and 'Set Ratio' (0.3). A blue button labeled 'Get Data From GraphQL' is positioned below the form. The results section, titled 'Keywords', contains the following text: 'takes valuable time, computer science courses, computer science studies, study computer science, computer science, aptitude test, admission-free subject, imparts knowledge, key competencies, lack self-motivation, study generally, secondary subject'.

Figure 5.19.: Keywords extracted using the following parameters: *positionID* = 320, *inputModel* = Rake, *ratio* = 0.3, *considerStructure* = false.

The screenshot shows a web browser window with the URL `127.0.0.1:8000/keywords/search/?user=test_user&positionID=320&inputModel=rake&considerStructure=false&ratio=0.6`. The page has a navigation bar with 'Home', 'Topics', 'Summary', 'Keywords', and 'API'. Below the navigation bar, there is a form with the following fields: 'User' (test_user), 'Position ID' (320), 'Choose Model' (Rake), 'Consider Structure' (False), and 'Set Ratio' (0.6). A blue button labeled 'Get Data From GraphQL' is positioned below the form. The results section, titled 'Keywords', contains the following text: 'takes valuable time, computer science courses, computer science studies, study computer science, computer science, admission-free subject, lack self-motivation, aptitude test, imparts knowledge, key competencies, study generally, secondary subject, suitability test, students fail, studies, study, suitability, secondary, students, determine, successful, long, prevent'.

Figure 5.20.: Keywords extracted using the following parameters: *positionID* = 320, *inputModel* = Rake, *ratio* = 0.6, *considerStructure* = false.

Changing the *ratio* parameter or choosing another value for the *inputModel* parameter will generate either another amount of keywords or another sets of keywords respectively. As seen

in Figure 5.20, increasing the *ratio* parameter leads to a larger set of generated keywords.

The choice of another value for the *inputModel* parameter leads to another set of extracted keywords. This example is illustrated in Figure 5.21 where a user set the *inputModel* parameter to the *TF* option.

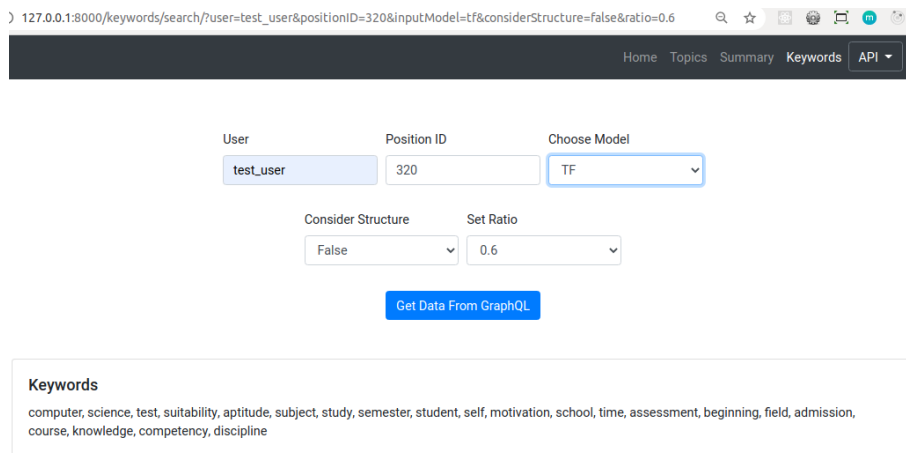


Figure 5.21.: Keywords extracted using the following parameters: *positionID* = 320, *inputModel* = TF, *ratio* = 0.6, *considerStructure* = false.

In case the invalid parameters are provided, the proposed keyword extractor generates the corresponding errors and displays them. One of the cases of the error handling is shown in Figure 5.22 where the url was changed and the *inputModel* parameter was set to a nonexistent value “testModel“. The invalid part of the url is marked in red.

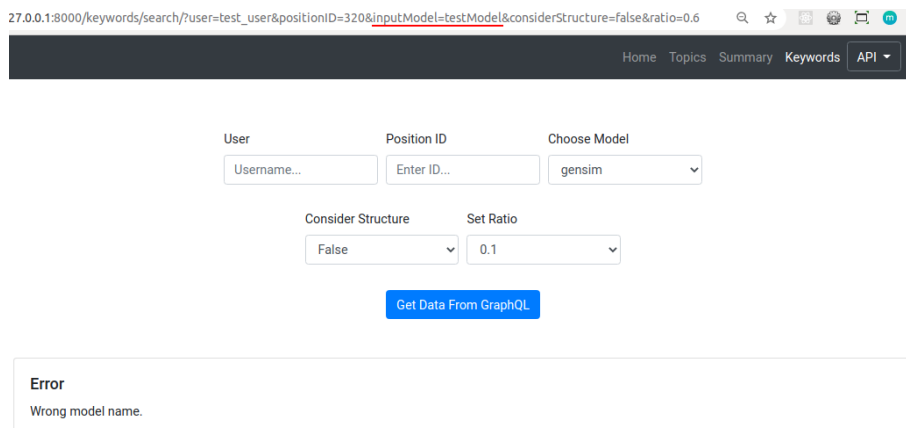


Figure 5.22.: Example of the error handling when an invalid *inputModel* parameter is provided.

5.4.3. Keyword Extraction

Similarly to the *summary* app, this module also enables users to decide if the position structure has to be taken into account. If the *considerStructure* parameter is set to *true*, the statements that have premises will be duplicated to give these statements an extra weight. Similar to the summarization process, the statements were duplicated only once to avoid the biasing of short statements that don't bear any significant information.

If the *inputModel* parameter was set to *gensim* or *rake*, the input data will be preprocessed internally by the algorithms provided by corresponding packages. If the *custom model* option is set, the input text is preprocessed in the same way as for text summarization. Additionally, all parts of speech except for nouns are removed, since usually nouns bear the most important information.

When *inputModel* is set to *custom model*, the proposed tool will use the *TF* approach as for the summarization process. The proposed tool creates a frequency table, where the frequency of the remaining tokens is tracked. In the last step, the terms with the highest scores are extracted. The amount of keywords is regulated by the provided *ratio* parameter.

By setting the *inputModel* parameter to *gensim*, the proposed tool utilizes the implemented algorithm from Subsection 4.4.3. It creates a unidirectional graph, in which edges are drawn between those tokens that co-occur within a predefined window size. After this step, the PageRank ranking algorithm is applied on the graph. In the last step, the keywords with the highest ranks are retrieved.

The *Rake* option triggers the Rake algorithm. For this algorithm, the minimum length of terms is set to 3 to filter the terms which do not bear any significant information. The maximum number of words per keywords is set to 3 for the positions in English and to 1 for the positions in German. This decision was made due to the fact that German sentences may have a specific word ordering which results in a poor performance of this algorithm. The minimum frequency of terms is set to 1, because the text of the positions can be very short and all terms can only co-occur once. After the input text is preprocessed and tokenized, a co-occurrence table is created. In the next step, the remaining terms are scored as described in Subsection 4.4.2. Finally, the top-ranked terms are extracted.

The extracted keywords are stored within the *Keywords* model defined in the *models.py* of

this module. This model has the same structure as *Topic* and *Summary* models.

5.4.4. API

The serialized data is structured in the same way as in the *Keywords* model. The *API* of this module provides two endpoints, which use *GET* request methods to retrieve the resources.

- */api/keywords/*: By providing the same parameters as in Subsection 5.4.1 to this endpoint, users can generate keywords and retrieve them from the database.
- */api/keywords/{pk}*: By providing a primary key parameter to this endpoint, keywords with the specified *primary key* can be obtained.

Figure 5.23 illustrates how users can make a request to the first endpoint.

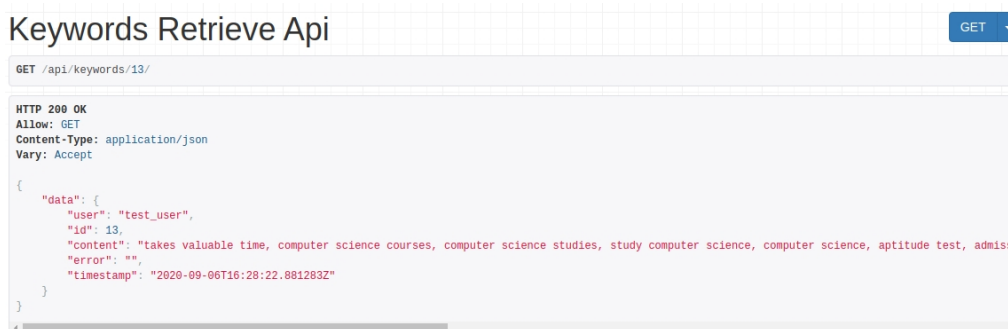


```
Keywords List Api GET
GET /api/keywords/?user=test_user&positionID=320&inputModel=tf&considerStructure=false&ratio=0.5

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": [
    {
      "user": "test_user",
      "id": 12,
      "content": "computer, science, study, test, subject, suitability, aptitude, way, semester, admission, course, knowledge, competency, discipline,
      "error": "",
      "timestamp": "2020-09-06T16:26:28.342716Z"
    }
  ]
}
```

Figure 5.23.: Keywords List API with the following parameters: *user* = *test_user*, *positionID* = *320*, *considerStructure* = *false*, *inputModel* = *rake*, *ratio* = *0.5*.



```
Keywords Retrieve Api GET
GET /api/keywords/13/

HTTP 200 OK
Allow: GET
Content-Type: application/json
Vary: Accept

{
  "data": {
    "user": "test_user",
    "id": 13,
    "content": "takes valuable time, computer science courses, computer science studies, study computer science, computer science, aptitude test, admissi
    "error": "",
    "timestamp": "2020-09-06T16:28:22.881283Z"
  }
}
```

Figure 5.24.: Keywords Retrieve API requesting a resource with the *primary key* = *160*.

The keywords of the position with the id = 320 are to be generated and retrieved. The keywords will be generated by the Rake algorithm with the ratio set to 0.5. The structure of the position will not be considered since the corresponding parameter is set to *false*.

In order to retrieve already existing keywords, the user can make a request to the second endpoint like on Figure 5.24. In this example, a resource with the id = 13 is requested.

Chapter 6.

Evaluation

This chapter introduces the evaluation results of the proposed topic extractor, summary generator and keywords extractor, as well as analyzes the influence of some factors on the accuracy of the generated results.

6.1. Topic Modeling

There are two approaches to evaluate topic models: to use statistical metrics, such as perplexity or coherence, or to use human evaluation. Perplexity measures how well a model predicts samples. Coherence measures a semantic similarity between high scoring terms in a topic. Though, in numerous cases statistical metrics prove to be reliable, in a context of natural language tasks such metrics are not informative since there is no ground truth which values of these statistical metrics can be compared to. As a consequence, the results of such metrics are not strongly correlated to human judgement. For this reason, the human evaluation was used to assess performance of the proposed topic extractor.

6.1.1. Evaluation Setup

A group of seven participants was asked to read two discussions: one discussion in German and the same discussion translated in English. For these discussions, several models were generated. Table 6.1 displays the parameter constellation that was used to generate models for evaluation. As it can be derived from the table, three model-sets (*LDA*, *NMF* and *LSI*) were created for the *sklearn* package and two model-sets (*LDA* and *LSI*) were created for the *gensim* package. Furthermore, each model-set contains three and five topics each of which were created with and without the usage of an externally pre-trained model. The *nGrams* parameter was set to use only unigrams since the size of the *D-BAS* dataset is currently not sufficient to build reasonable n-grams.

Model	NMF	LDA	LSI
Package	sklearn	sklearn, gensim	sklearn, gensim
Pre-trained model	false, true	false, true	false, true
Num. topics	3, 5	3, 5	3, 5
N-grams	false	false	false

Table 6.1.: Topic parameters for evaluation.

After the participants finished reading both discussions, they were asked to fill out two questionnaires. Every task in the questionnaire includes the above described topics extracted from the corresponding discussion followed by a scale from 1 to 10, where 1 stands for “Doesn’t fit at all” and 10 stands for “Fits perfectly“. Using this scale, the participants had to rank the accuracy of the given topics.

An example of such a task is illustrated in Figure 6.1. In this task, the user is asked to evaluate the *gensim LDA* model consisting of three topics that were generated without the usage of an externally pre-trained model.

Apart from the questionnaire tasks described above, another additional task which included all the topics from the pre-trained models. For a full list of topics, see Appendices A, B, C, D. The participants were asked to select the topics which they consider to be relevant for the corresponding discussions. An additional option “None of them“ was included into these topic lists.

The generation of the model-sets with and without pre-trained models have a particular impact on the results. Therefore, their evaluation will be discussed in the subsequent sections

Topic Modeling without a pretrained model: gensim, LDA (3 topics)

Topic 1: student, computer, study, course, question, high, science, scientist, content, learn, pass, require, time, solution, discuss

Topic 2: lecture, student, computer, study, science, subject, time, mathematic, require, module, exercise, course, number, compulsory, mathematical

Topic 3: student, computer, lecture, science, course, exercise, subject, study, school, task, good, mathematic, seminar, sheet, solution

7. How relevant do these topics sound for you? (Can you distinguish between topics?)

1 2 3 4 5 6 7 8 9 10

Doesn't fit at all Fits perfectly

Figure 6.1.: An example exercise to evaluate an LDA model of the *gensim* package with three topics.

separately.

6.1.2. Topics with Pre-trained Models

For this part of the evaluation, the pre-trained models described in Subsection 5.2.4 were used to generate the topics. The datasets and the parameters used to pre-train these models were also described in the same section.

Figure 6.2 shows that all participants uniformly chose one single topic, which is the most fitting for the domain of this particular discussion. These are the 20 most significant tokens within this topic: *university, school, college, education, year, program, research, institute, national, study, high, society, campus, center, member, work, hospital, academy, graduate, science*.

This single topic was chosen due to the fact that the external dataset for the pre-trained model represents general domain knowledge, while this particular *D-BAS* discussion is encapsulated within a very specific domain, which deals not with the university life in general, but with the improvement of the computer science course in particular. For this reason, no further

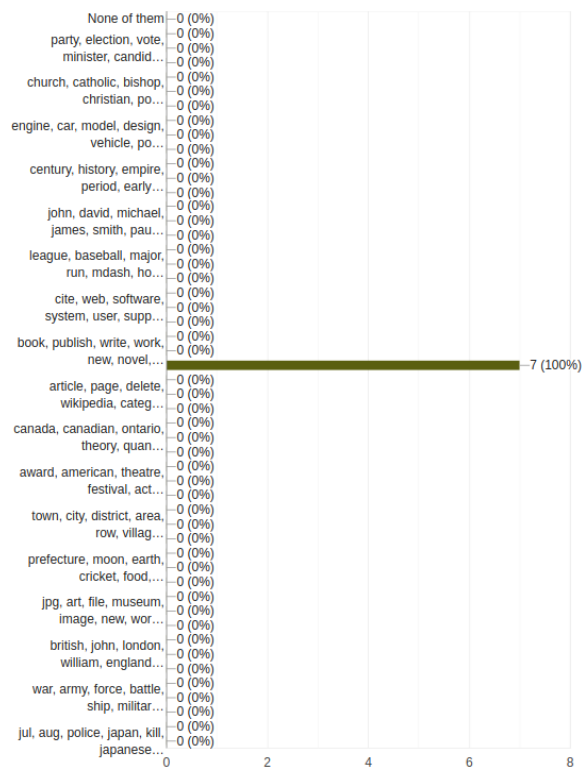


Figure 6.2.: An example task to evaluate the pre-trained *gensim* LDA model with 50 topics (English, *gensim*).

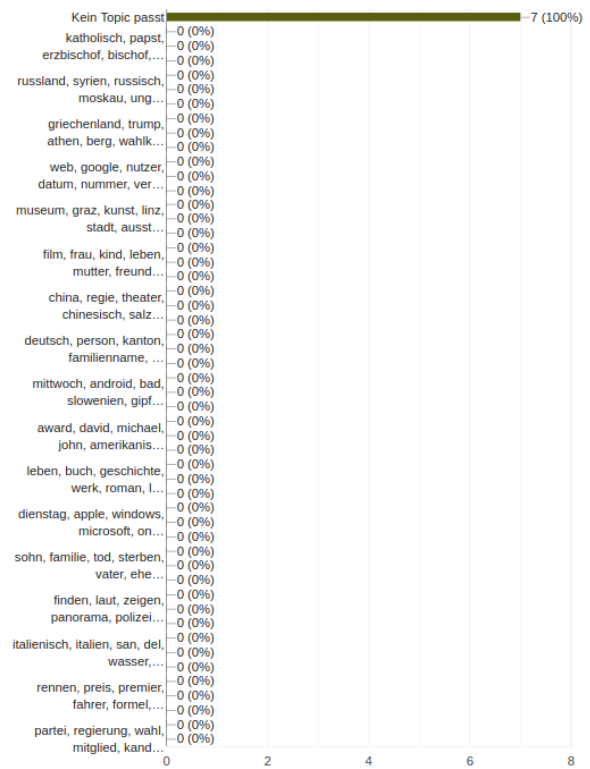


Figure 6.3.: An example task to evaluate the pre-trained *gensim* LDA model with 50 topics (German, *gensim*).

fine-grained topics could be selected by the participants.

In comparison to the English discussion, the evaluation of the results for the German discussion revealed that all participants found all 50 externally pre-trained topics irrelevant for the original discussion and chose the option “Kein Topic passt“ (None of them) uniformly. Figure 6.3 visualizes these results.

If one considers these topics, which are listed in Appendix C explicitly, one notices that none of the topics covers the domain of the German discussion at least generally. This proves once more that the usage of external datasets for the purposes of *D-BAS* topic extractor does not improve the accuracy of the extracted topics.

Consider Figures 6.4, 6.5, 6.6, 6.7, and 6.8 which display the overall results of the evaluation of the topics with pre-trained models. The highest score given for these topics is 3 out of 10. In some cases, all participants uniformly ranked the performance of the models with the lowest possible score. Such poor results support the idea that external datasets are incapable

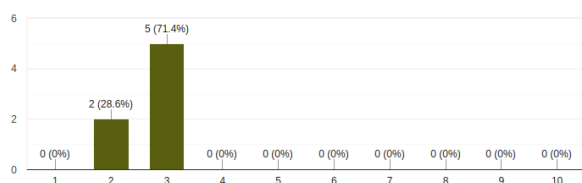


Figure 6.4.: LDA — five topics with a pre-trained model (English, *gensim*).

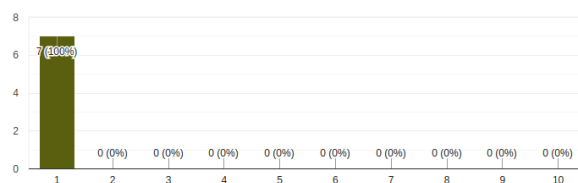


Figure 6.5.: LSI — five topics with a pre-trained model (English, *gensim*).

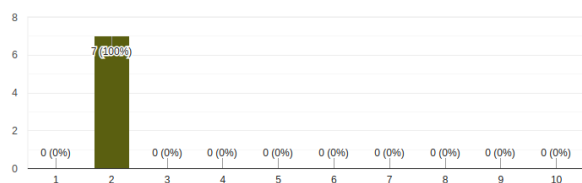


Figure 6.6.: NMF — five topics with a pre-trained model (English, *sklearn*).

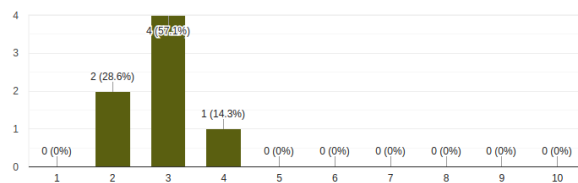


Figure 6.7.: LDA — five topics with a pre-trained model (English, *sklearn*).

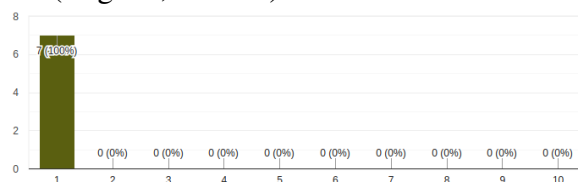


Figure 6.8.: LSI — five topics with a pre-trained model (English, *sklearn*). On these scales, 1 stands for “Doesn’t fit at all“ and 10 stands for “Fits perfectly”.

of capturing a domain specific nature of the *D-BAS* discussions.

The same tendency can be observed in Figures 6.9, 6.10, 6.11, 6.12, and 6.13, where the results of the evaluation for the German discussion are illustrated. All model-sets received the lowest amount of points. Though, comparing results for both discussions, it is obvious that the models for the German discussion perform worse than the models for the English discussion.

One of the reasons why the German models delivered worse results is the fact that German and English external datasets do not have any one-to-one mapping between each other. In other words, it is not the case that the German dataset was translated into English or vice versa, and both datasets are completely independent of each other. The English dataset contains general information from the English discussion (university and education specific domains), but the German dataset does not contain any data which could cover the domain of

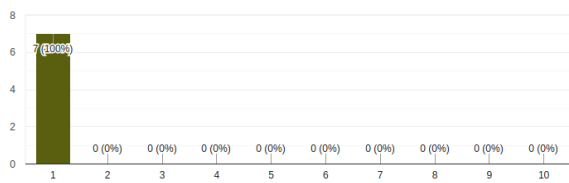


Figure 6.9.: LDA — five topics with a pre-trained model (German, *gensim*).

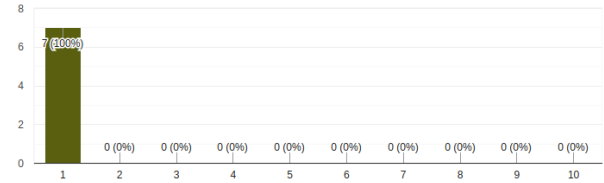


Figure 6.10.: LSI — five topics with a pre-trained model (German, *gensim*).

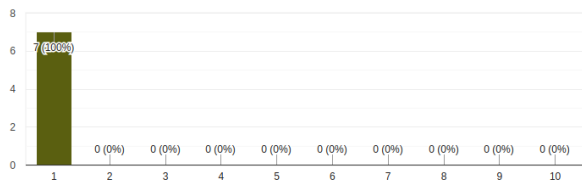


Figure 6.11.: NMF — five topics with a pre-trained model (German, *sklearn*).

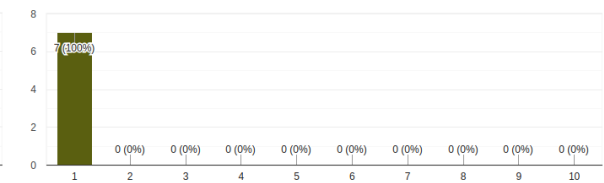


Figure 6.12.: LDA — five topics with a pre-trained model (German, *sklearn*).

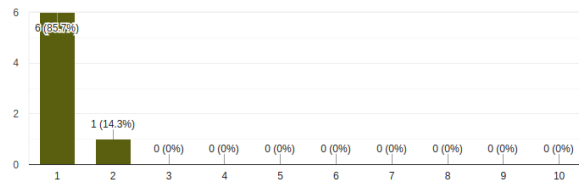


Figure 6.13.: LSI — five topics with a pre-trained model (German, *sklearn*).

the German discussion.

Another explanation of such poor results of the German model-sets are the peculiarities of German language itself. Particularly, the word ordering within sentences and the German composite words present a problem. For example, the German terms such as “*Informatikstudium*“, “*Informatik-Studium*“, “*Informatik Studium*“ represent the same concept and should be considered as a single term by a topic model to deliver a better performance.

6.1.3. Topics without Pre-trained Models

The evaluation of the topics without pre-trained models provides the following results. Figure 6.14 and Figure 6.15 display the result distributions for the *LDA* and *LSI* topic models of the

gensim package. In 85,7% of the responses, which corresponds to six of seven participants, the *LDA* topics received 5 points out of possible 10. In the same amount of responses, the *LSI* model received 4 points out of 10. Only one participant ranked these topics with 4 and 2 points respectively.

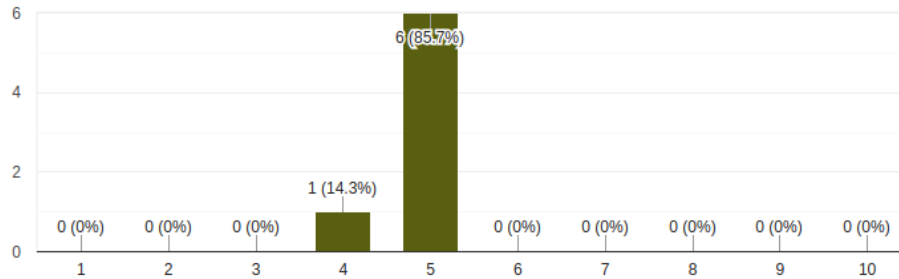


Figure 6.14.: LDA — five topics without a pre-trained model (English, *gensim*).

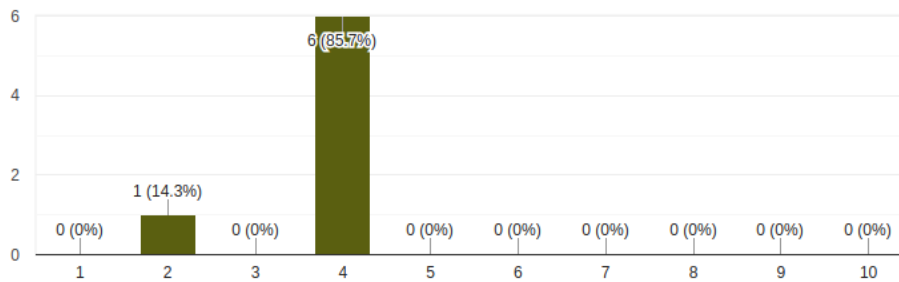


Figure 6.15.: LSI — five topics without a pre-trained model (English, *gensim*).

Similarly, Figure 6.16, 6.17, and 6.18 illustrate result distributions for the *NMF*, *LDA*, and *LSI* models of the *sklearn* package. In 71.4% of the responses, the *NMF* model was ranked with 5 points of possible 10 and only two participants ranked these topics with 6 points.

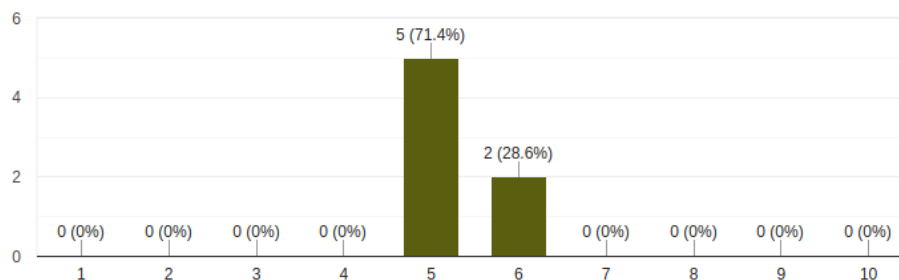


Figure 6.16.: NMF — five topics without a pre-trained model (English, *sklearn*).

The *sklearn LDA* model received 5 points of possible 10 in 85.7% of all the cases. The *LSI* model was ranked with 4 points by six of seven participants.

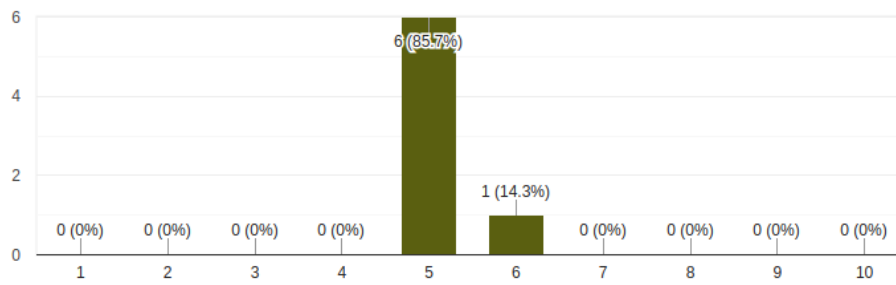


Figure 6.17.: LDA — five topics without a pre-trained model (English, *sklearn*).

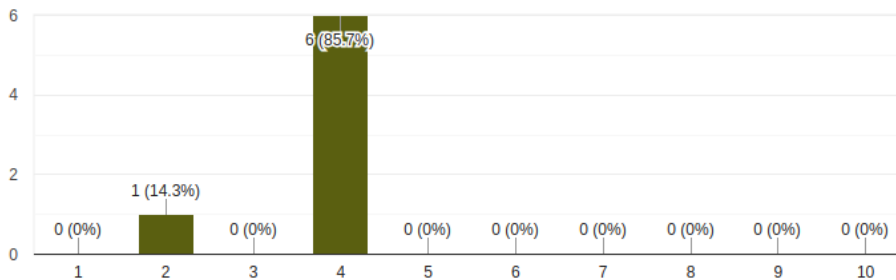


Figure 6.18.: LSI — five topics without a pre-trained model (English, *sklearn*).

Given the above described figures, both *LDA* and *NMF* models perform almost equally. The overall performance of the *LSI* model is lower.

Comparing the evaluation results of the English model-sets with the evaluation results of the German models, it is evident that both German and English models perform almost equally. Figures 6.19 and 6.20 display the result distributions of the evaluation. As in case of the English discussion, the German *LDA* model of the *gensim* received 5 points in 85.7% of all responses and only one participant ranked this model with 4 points.

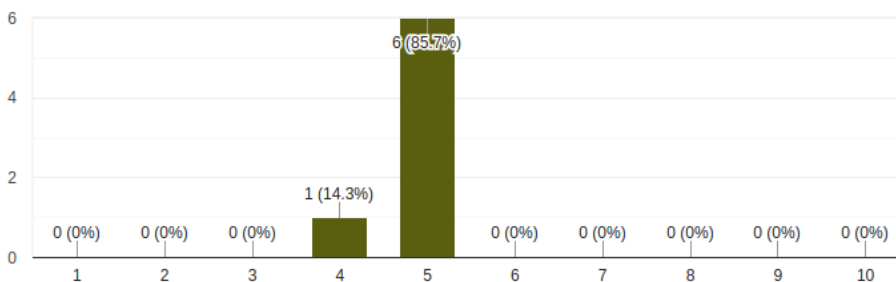


Figure 6.19.: LDA — five topics without a pre-trained model (German, *gensim*).

As in case of the English *LSI* model of the *gensim* package, the German *LSI* model performed worse than other models and received between 3 and 4 points of possible 10.

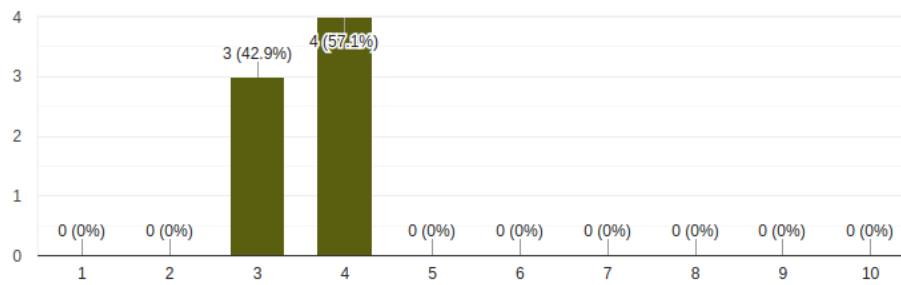


Figure 6.20.: LSI — five topics without a pre-trained model (German, *gensim*).

The German *NMF* and *LSI* models of the *sklearn* package received average ranking similarly to the models of the *gensim* package which becomes clear from Figure 6.21 and Figure 6.23. The *LDA* model, on the contrary, received a lower ranking (3 points of possible 10 in 85.7% of responses). Such results can be caused by the German language peculiarities discussed in Subsection 6.1.2.

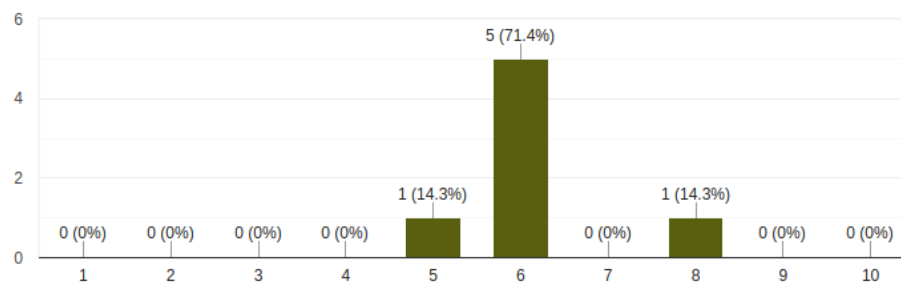


Figure 6.21.: NMF — five topics without a pre-trained model (German, *sklearn*).

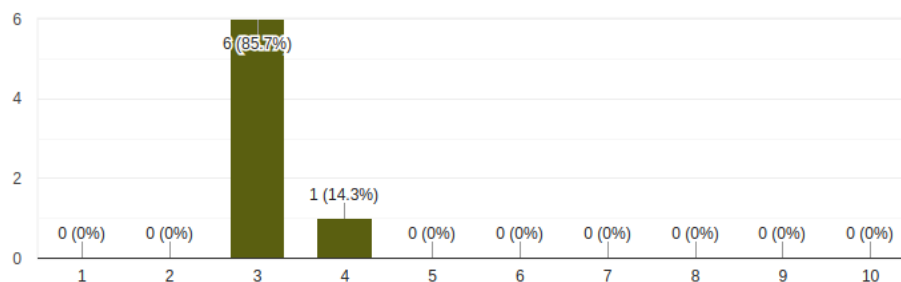


Figure 6.22.: LDA — five topics without a pre-trained model (German, *sklearn*).

Such an average performance of the provided models is due to sparse *D-BAS* data, which results in the overlapping of the topics. That is why a clear distinction between these topics cannot be provided. These results suggest that the proposed topic extractor provides users with relevant topics that are sufficiently performant to give an overview of the information hidden in the discussion, but it should be enhanced with further improvements to achieve a higher accuracy.

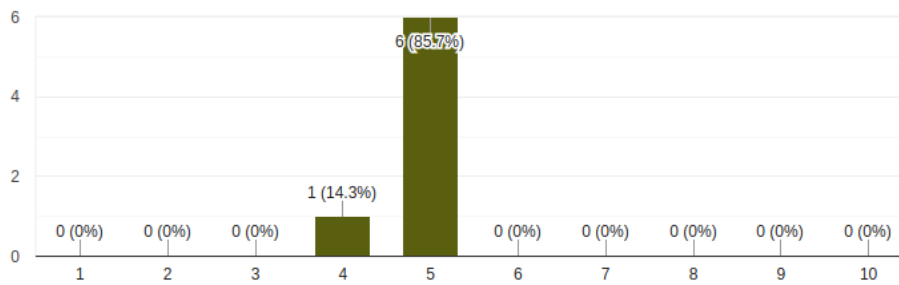


Figure 6.23.: LSI — five topics without a pre-trained model (German, sklearn).

6.2. Text Summarization

As in case of the above explained evaluation, the automatically generated summaries can be evaluated by two major approaches: mathematical metrics or human evaluation. First, the results of the human evaluation will be explained. Subsequently, a metric such as a *ROUGE* score will be applied to the generated summaries. A short explanation of this metric will be given in Subsection 6.2.3.

6.2.1. Evaluation Setup

For the evaluation of the summary extraction, 15 participants were asked to read two positions from the discussions used for the evaluation of the topic extractor. All summaries were generated with *ratio* = 0.5 without considering a position structure. The position from the English discussion is shown in Figure 6.24 and the same position from the German discussion is given in Figure 6.25.

1. I do not see any other way to determine whether one is able to be successful at computer science studies
2. studying not only imparts knowledge, but also key competencies, which may be important in other disciplines
3. one can determine the suitability for the study generally only during the study. Many students fail because they lack self-motivation, which is secondary in during school
4. it would be helpful for students to get an assessment at the beginning. You could design the aptitude test such that students can still start the study
5. this will be shown in the first semesters anyway
6. as long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses
7. then students, who are not ready to study in the field of computer science, can not start the studies
8. I basically do not see the need for an aptitude test
9. one takes valuable time from the students by this

Figure 6.24.: The position (*positionID* = 320) from the *D-BAS* discussion in English which was later used to extract summaries for the evaluation step.

After the participants read these positions, they had to rank several summaries generated by the algorithms explained in Subsection 5.3.4. These summaries are provided in Appendix E.

1. ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen
2. solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird
3. ich grundsätzlich die Notwendigkeit einer Eignungsprüfung nicht sehe
4. man damit den Studenten wertvolle Zeit wegnimmt
5. es für Studierende hilfreich wäre, vorher eine Einschätzung zu bekommen. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten
6. sich das ohnehin in den ersten Semester zeigt
7. es fast immer eine andere Möglichkeit/Lösung gibt
8. man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. Viele Studenten scheitern, weil ihnen die Selbstmotivation fehlt, was in der Schule eher zweitrangig ist
9. so Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können
10. das Studium nicht nur Fachwissen, sondern auch Schlüsselkompetenzen vermittelt, die in anderen Disziplinen wichtig sein können

Figure 6.25.: The position (*positionID* = 50) from the *D-BAS* discussion in German which was later used to extract summaries for the evaluation step.

Every task consisted of a generated summary and a scale from 1 to 5 where participants could rank the results. An example task is presented in Figure 6.26. In this example, the summary generated by the *gensim* package is to be evaluated.

1. Summary: gensim approach

I do not see any other way to determine whether one is able to be successfull at computer science studies. then students, who are not ready to study in the field of computer science, can not start the studies. one can determine the suitability for the study generally only during the study. You could design the aptitude test such that students can still start the study. as long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses

	1	2	3	4	5	
Doesn't fit at all	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	Fits perfectly

Figure 6.26.: An example task to rank the summary produced by the *gensim* package (English).

Additionally, the participants could give their own summary of the position. These summaries will later be used as reference summaries to compute the *ROUGE* scores of the generated summaries.

6.2.2. Human Evaluation

Considering the summaries generated by the *gensim* package, 80% of the participants ranked the English summary with 5 points which is the highest possible score. The German summary received 4 points in 66.7% of the responses, two participants ranked the accuracy of this summary with 5 points. It is obvious that the summarization of the English position performed better in comparison to the summarization of the German position. This resulted by the difference of the internal pre-processing of the text within the package. Particularly,

if no German model is used internally to clean the data, then the meaningless terms such as stopwords will also be taken into account.

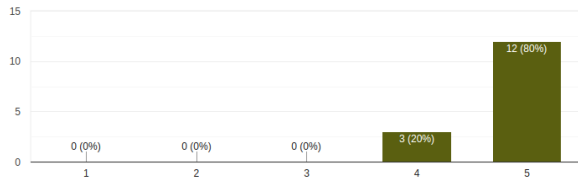


Figure 6.27.: Evaluation results of the summary generated by the *gensim* package (English).

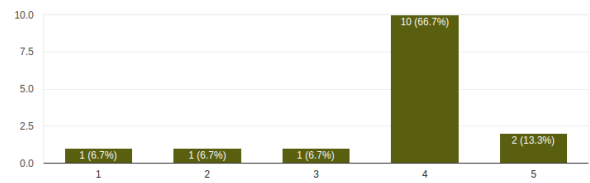


Figure 6.28.: Evaluation results of the summary generated by the *gensim* package (German).

The summaries produced by the *TextRank* algorithm of the *spacy* package delivered low rankings. 66.7% of the participants scored the English summary with 4 points of possible 5, four participants ranked this summary with 5 points. Figure 6.29 illustrate the result distribution for this summary.

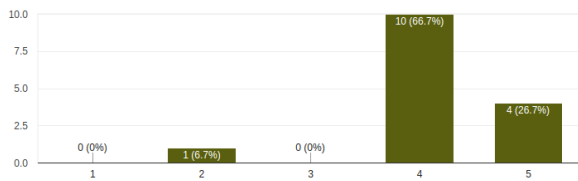


Figure 6.29.: Evaluation results of the summary generated by the *TextRank* algorithm (English).

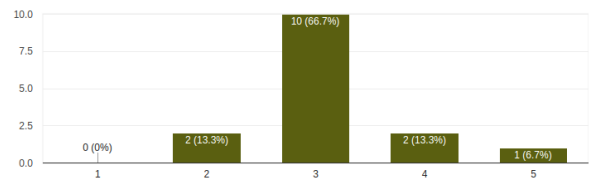


Figure 6.30.: Evaluation results of the summary generated by the *TextRank* algorithm (German).

The German summary received lower ranking in comparison to the English one. Figure 6.30 show the results of this evaluation. This summary received 3 points from ten participants, other rankings differ from 2 to 5 scores. This makes this summary to be less accurate than the German *gensim* summary.

Given the evaluation results of the summary generated by the custom *TF* algorithm displayed in Figures 6.31 and 6.32, it is obvious that the English summary showed a higher accuracy than the German one. The English summary received 4 points of possible 5 from 80% of the participants whereas the German summary was ranked 3 points in 73.3% of responses. Yet again, the difference in accuracy may be caused by the language specificity of the German language which was already discussed in Subsection 6.1.2.

It is clear from Figures 6.33 and 6.34 that the summaries, generated by the *TF-IDF* algorithm, received the average rankings. The majority of the participants (66.7%) ranked the

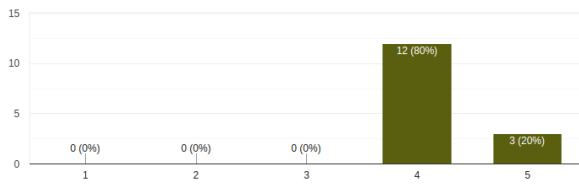


Figure 6.31.: Evaluation results of the summary generated by the *TF* algorithm (English).

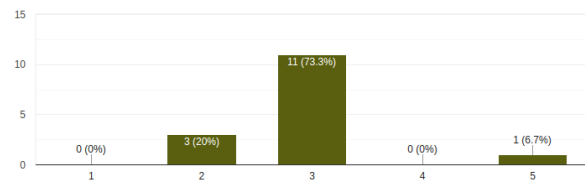


Figure 6.32.: Evaluation results of the summary generated by the *TF* algorithm (German).

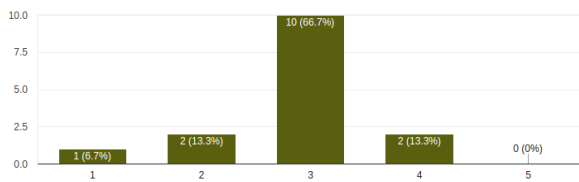


Figure 6.33.: Evaluation results of the summary generated by the *TF-IDF* algorithm (English).

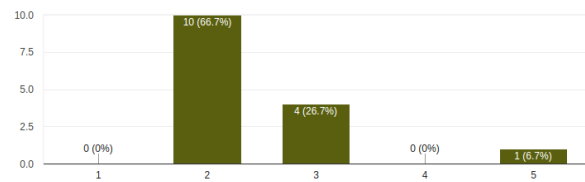


Figure 6.34.: Evaluation results of the summary generated by the *TF-IDF* algorithm (German).

English summary with 3 points. Ten participants gave 2 points to the German summary, four participants ranked it with 3 points.

To sum it up, the *gensim* and *spacy TextRank* summaries showed higher accuracy than other frequency-based summaries due to the fact that the *TextRank* algorithm, which underlies both packages and is able to capture the contextual connection between the sentences. Finally, the German summaries proved to be less accurate in comparison to the English ones due to the peculiarities of the German language and a possible absence of the stopwords removal process within the packages.

6.2.3. ROUGE Scores

ROUGE stands for *Recall-Oriented Understudy for Gisting Evaluation*. It compares an automatically generated summary with a set of human-produced reference summaries. *ROUGE* deals with such terms as *precision* and *recall*. *Recall* is computed as shown in Equation 6.1. The recall measures how good a generated summary captured words from a reference summary.

$$Recall = \frac{\text{number of overlapping words}}{\text{total number of words in reference summary}} \quad (6.1)$$

However, it can be the case that the summary captures the most part of the original words but is unnecessarily long making a lot of the words useless. *Precision* measures how relevant the generated summary is and is calculated as shown in Equation 6.2.

$$Precision = \frac{\text{number of overlapping words}}{\text{total number of words in generated summary}} \quad (6.2)$$

ROUGE-N metric refers to the overlapping of n-grams between the generated summaries and reference summaries. For instance, *ROUGE-1* only considers overlapping of unigrams and *ROUGE-2* measures the matching of bi-grams. *ROUGE-L* measures the longest overlapping sequence of words. *ROUGE-SU4* uses bigrams with a maximum skip distance of 4 between them.

	gensim	TextRank	TF	TF-IDF
<i>ROUGE-1</i> _{Recall}	0.4786	0.4202	0.4838	0.4283
<i>ROUGE-1</i> _{Precision}	0.2458	0.2125	0.2326	0.3013
<i>ROUGE-2</i> _{Recall}	0.1709	0.112	0.1709	0.1012
<i>ROUGE-2</i> _{Precision}	0.0897	0.0641	0.0833	0.06
<i>ROUGE-L</i> _{Recall}	0.3943	0.2848	0.3673	0.2589
<i>ROUGE-L</i> _{Precision}	0.2	0.1458	0.1705	0.1923
<i>ROUGE-SU4</i> _{Recall}	0.2410	0.1672	0.2456	0.1595
<i>ROUGE-SU4</i> _{Precision}	0.1146	0.0811	0.1095	0.1048

Table 6.2.: Average ROUGE scores for generated English summaries.

For the performance evaluation of the proposed summary generator, the recall and precision for uni- and bigrams as well as for the longest overlapping sequence were calculated. Additionally, the recall and precision for the bigrams with respect to the possible skip distance of 4 was examined. Both generated and reference summaries were pre-processed before computing the scores to remove redundant information which could bias the scores. Table 6.2 illustrates the *ROUGE* scores computed for the generated summaries. The highest scores are highlighted in bold print. All reference summaries are listed in Appendix E.9.

Considering the above listed results, it is clear that in all the cases the precision of the generated summaries is lower than the recall. For all four techniques which were used for the summary generation, the recall of unigrams is higher than the recall of the bigrams and the longest overlapping sequences. This is due to the fact that it is more probable to find overlapping unigrams in both generated and reference texts than the overlapping n -grams with $n > 1$.

	gensim	TextRank	TF	TF-IDF
<i>ROUGE</i> – 1_{Recall}	0.4786	0.4202	0.612	0.4283
<i>ROUGE</i> – $1_{Precision}$	0.2458	0.2125	0.25	0.3013
<i>ROUGE</i> – 2_{Recall}	0.1709	0.112	0.2132	0.1012
<i>ROUGE</i> – $2_{Precision}$	0.0897	0.0641	0.0851	0.06
<i>ROUGE</i> – L_{Recall}	0.3943	0.2848	0.4328	0.2589
<i>ROUGE</i> – $L_{Precision}$	0.2	0.1458	0.177	0.1923
<i>ROUGE</i> – $SU4_{Recall}$	0.2410	0.1672	0.2746	0.1669
<i>ROUGE</i> – $SU4_{Precision}$	0.1146	0.0811	0.1095	0.11

Table 6.3.: Average ROUGE scores for generated English summaries considering the position structure.

Additionally, these results show that from the mathematical point of view the overall performance of the introduced extractive summary generation techniques is below the average.

It is worth noting that according to the above mentioned results, the *gensim* summary stands out in performance in comparison to other summaries. This fact, in its turn, reflects the results of the human evaluation, in which the *gensim* summary received the best rankings in comparison to the summaries produced by other techniques.

Positive changes in the results were noticed while evaluating the summaries that were generated with the same parameters but with taking the position structure into account. Figure 6.3 illustrates the average *ROUGE* scores for these summaries. The *TF* approach showed both higher recall and precision in comparison to the results in Table 6.2. As expected, the *gensim* and *TextRank* approaches did not perform better since they work on a sentence level. Unexpectedly, the *TF-IDF* technique did not show any significant changes in the performance, though it was awaited that the repetition of particular sentences in the input text would lead to higher scores for some tokens. Given these results, it is justified to say that the *TF* approach slightly outperformed the *gensim* technique.

	gensim	TextRank	TF	TF-IDF
<i>ROUGE</i> – 1 _{Recall}	0.2622	0.2307	0.3047	0.2059
<i>ROUGE</i> – 1 _{Precision}	0.1538	0.1324	0.1458	0.1473
<i>ROUGE</i> – 2 _{Recall}	0.0228	0.0228	0.0507	0.0279
<i>ROUGE</i> – 2 _{Precision}	0.015	0.0143	0.0243	0.0184
<i>ROUGE</i> – L _{Recall}	0.1673	0.1878	0.2121	0.1377
<i>ROUGE</i> – L _{Precision}	0.0952	0.0976	0.0982	0.0892
<i>ROUGE</i> – SU4 _{Recall}	0.0944	0.0889	0.102	0.0535
<i>ROUGE</i> – SU4 _{Precision}	0.0518	0.0466	0.0457	0.0414

Table 6.4.: Average *ROUGE* scores for generated German summaries.

Table 6.4 demonstrates the *ROUGE* scores for the German summaries. Similar to the *ROUGE* scores for the English summaries, the scores for the German summaries are not only below the average in general but also below the scores for the English summaries in particular. Such an effect might be due to the peculiarities of the German language such as word ordering and

composite words.

Interestingly, in comparison to the results of the English summaries, there is one approach whose accuracy prevails all other techniques in almost all cases — the *TF* approach. Though, according to the human evaluation this technique displayed the average performance.

Considering the position structure and evaluating the generated summaries, it is worth mentioning that both *TF* and *TF-IDF* approaches delivered results with lower accuracy than in case of the English summaries that took the position structure into account. Table 6.5 illustrates the computed *ROUGE* scores for these summaries. As seen in this table, the *TextRank* and *gensim* approaches, retained the same scores as in Table 6.4. This was an expected behaviour because both approaches build graphs based on the whole sentences and not on the single tokens. For this reason, the repetition of the sentences would not lead to any changes in weightings.

	gensim	TextRank	TF	TF-IDF
<i>ROUGE</i> – 1_{Recall}	0.2622	0.2307	0.25	0.2059
<i>ROUGE</i> – $1_{Precision}$	0.1538	0.1324	0.1429	0.1473
<i>ROUGE</i> – 2_{Recall}	0.0228	0.0228	0.0	0.0279
<i>ROUGE</i> – $2_{Precision}$	0.015	0.0143	0.0	0.0184
<i>ROUGE</i> – L_{Recall}	0.1673	0.1878	0.1456	0.148
<i>ROUGE</i> – $L_{Precision}$	0.0952	0.0976	0.0806	0.0982
<i>ROUGE</i> – $SU4_{Recall}$	0.0944	0.0889	0.0779	0.0477
<i>ROUGE</i> – $SU4_{Precision}$	0.0518	0.0466	0.0419	0.0373

Table 6.5.: Average ROUGE scores for generated German summaries considering the position structure.

The results from Tables 6.3 and 6.5 lead to the conclusion that taking into the consideration the position structure does not necessarily improve the accuracy of the applied approaches.

Finally, comparing the results of the human evaluation with the results from Tables 6.2, 6.4, 6.3 and 6.5, it is obvious that the former do not correlate with the results provided by the *ROUGE* metric.

6.3. Keyword extraction

As in case of the summary extractor, the accuracy of the extracted keywords can be measured either by applying a mathematical metric or by means of human evaluation. The results of the human evaluation will be presented first. Subsequently, the *ROUGE* scores of the extracted keywords will be discussed.

6.3.1. Evaluation Setup

For the evaluation of the proposed keyword extractor, 15 participants were asked to read the same positions as in Subsection 6.2.1. After this step, they had to evaluate the accuracy of the extracted keywords by ranking them on a scale from 1 to 5 where 5 is the highest possible score.

The generated keywords are listed in Table F.1 of Appendix F. All keywords were generated without considering a position structure with ratio = 0.5. Figure 6.35 illustrates an example task, in which the participants were asked to evaluate the accuracy of the keywords that were generated by the *Rake* algorithm.

8. Keywords: Rake approach

takes valuable time, computer science courses, computer science studies, study computer science, computer science, admission-free subject, imparts knowledge, key competencies, aptitude test, lack self-motivation, study generally, secondary subject

1 2 3 4 5

Doesn't fit at all Fits perfectly

Figure 6.35.: An example task to rank the keywords generated by the *Rake* algorithm (English).

Subsequently, the participants were asked to list the words which they consider to be most

relevant for the given positions. These keywords will later be used as reference keywords to compute the *ROUGE* scores.

6.3.2. Human Evaluation

From the result distribution of the keywords generated by the *gensim* package and displayed in Figures 6.36 and 6.37, it is clear that this approach delivered poor results for both the English and German positions. Though, the English keywords received slightly higher rankings: 11 participants ranked them with 1 point, two participants ranked them with 2 points, and 1 participant gave them 3 points. At the same time, 86.7% of all participants ranked the German keywords with 1 point of possible 5, and two participants ranked them with 2 points.

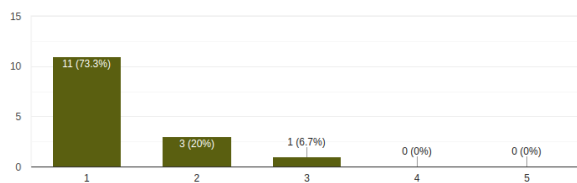


Figure 6.36.: Evaluation results of the keywords generated by the *gensim* package (English).

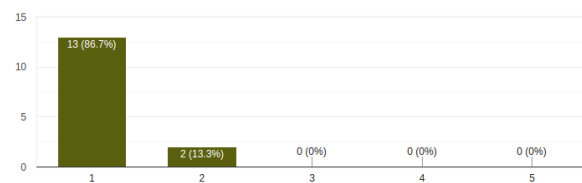


Figure 6.37.: Evaluation results of the keywords generated by the *gensim* package (German).

In comparison to the above mentioned keywords, the keywords generated by the *Rake* algorithm showed higher accuracy, which is illustrated in Figure 6.38. 60% of the participants ranked the keywords for the English discussion with 4 points, 33.3% of all participants gave 5 points, and only one participant ranked these keywords with 3 points. The situation looks differently for the position in German, which is shown in Figure 6.39. In 80% of all responses, the German keywords received the lowest scores — 1 point, and only 20% of the participants scored these keywords with 2 points.

In general, both externally implemented algorithms performed worse for the text in German than in English. Considering the German keywords generated by the *gensim* package, it is obvious that the main reason for the deficient performance of this package is the absence of the stop words removal process for texts in German. The reason for the insufficient performance of the *Rake* algorithm for the German text might be a peculiar word ordering of the German language.

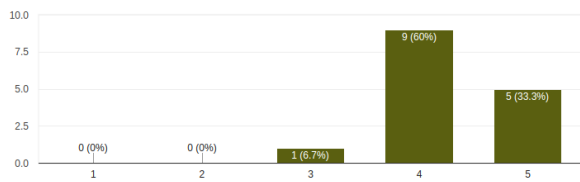


Figure 6.38.: Evaluation results of the keywords generated by the *Rake* algorithm (English).

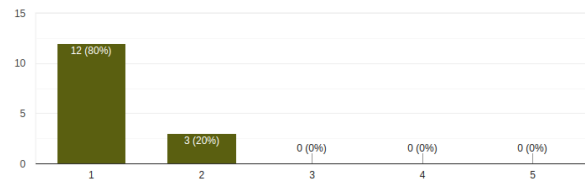


Figure 6.39.: Evaluation results of the keywords generated by the *Rake* algorithm (German).

Considering the evaluation results of the keywords for the position in English and generated by the custom *TF* algorithm, it is obvious that the majority of the participants (ten of 15) ranked them with 3 points, two participants gave them 2 and 4 points, and only one participant ranked them with 1 point.

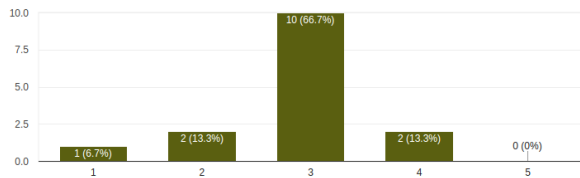


Figure 6.40.: Evaluation results of the keywords generated by the *TF* algorithm (English).

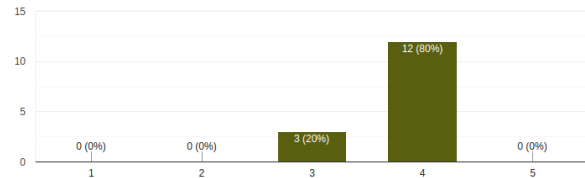


Figure 6.41.: Evaluation results of the keywords generated by the *TF* algorithm (German).

Compared to the above described results, the custom *TF* algorithm delivered the keywords for the position in German with higher accuracy. This is due to the fact that this approach cleans the data before the keywords are extracted. Figure 6.41 illustrates the result distribution for the German keywords.

One of the reasons why this approach was not evaluated with the highest scores might be the amount and the grammatical form of the extracted keywords. After the evaluation results were gathered, this algorithm was improved to take into account only the nouns since they usually carry the most important information. This reduces the amount of the generated keywords and improves their accuracy.

The above explained evaluation showed that the proposed keyword extractor is able to provide reasonable keywords from the discussion positions. Particularly, the custom *TF* approach is more stable for both languages in comparison to the externally implemented algorithms.

6.3.3. ROUGE Scores

To calculate the *ROUGE* scores for the generated keywords, each keyword/keyword phrase was considered to be a unigram. The amount of the unigrams occurring both in the extracted and the reference keywords was counted and then divided by the number of extracted keywords to calculate the recall, or by the number of keywords in the reference keywords to calculate the precision. The results for the English keywords are listed in Table 6.6. All reference keywords can be found in Table F.2 of Appendix F.

	gensim	Rake	TF	TF (improved)
$ROUGE - 1_{Recall}$	0.0714	0.3571	0.4286	0.4286
$ROUGE - 1_{Precision}$	0.1667	0.4167	0.1875	0.2609

Table 6.6.: Average ROUGE scores for generated English keywords.

Considering these results, the precision of the *Rake* approach received the highest scores. This coincides with the results of the human evaluation, in which the participants ranked the English keywords produced by the *Rake* algorithm with the highest scores.

The *ROUGE* scores of the German keywords are given in Table 6.7. Similarly to the results of the human evaluation, the German keywords extracted by the *Rake* received the lowest scores. At the same time, according to the *ROUGE* scores, the *TF* technique showed higher accuracy. These results coincide with the results of the human evaluation of this approach.

	gensim	Rake	TF	TF (improved)
$ROUGE - 1_{Recall}$	0.2273	0.0	0.5455	0.545
$ROUGE - 1_{Precision}$	0.2	0.0	0.3871	0.5714

Table 6.7.: Average ROUGE scores for generated German keywords.

In addition, it should be mentioned that the improved *TF* technique shows better precision, which supports the fact that considering only the nouns as keywords improves the performance of the algorithm.

Considering the *ROUGE* scores for the keywords in both languages, all applied techniques showed the accuracy below the average. Finally, it is worth mentioning that the overall *ROUGE* results do not correspond to the results of the human evaluation.

Chapter 7.

Conclusion

Within the framework of this master's thesis, an attempt was made to develop a tool which would combine a topic extractor, a summary generator and a keyword extractor for the *D-BAS* discussions. This tool should enable the participants of the discussions to get an overview on the most crucial information within a discussion by reducing its amount to the minimum and preserving the main facts.

The evaluation results showed that the proposed tool manages to reveal the hidden topics within a discussion without a usage of any externally pre-trained datasets. The insufficient performance of the pre-trained models supports the idea that external datasets are incapable of capturing a domain specific nature of the *D-BAS* discussions. Though, a current lack of the data within the *D-BAS* makes the accuracy of the topics, generated by the proposed topic extractor, to be average or below the average. The usage of the external datasets failed to improve the topic accuracy.

In the future, the proposed topic extractor could be enhanced with the following improvement. To enlarge the amount of the training data for the models, the external datasets could be combined with the accumulated *D-BAS* discussions data to pre-train the model and later to classify new discussion statements. These models can later be re-trained every time some amount of new data is accumulated. Such an improvement should capture the domain-specific nature of the *D-BAS* discussions. This topic extractor can also be enhanced by introducing new input parameters such as the learning rate, decay or batches, which would give the users more control over the models and might improve the accuracy of the generated topics.

The proposed summary and keyword extractor also fulfills its task and manages to provide the reasonable results, though with the average accuracy. This is due to the extractive nature of the applied approaches which suffer from the well-known problems such as a lack of balance and a lack of coherence. In the future, the proposed summary and keyword extractor can be improved by introducing the options that would allow the users to generate an abstractive summary. For example, the API introduced by OpenAI and mentioned in Subsection 4.3.4 can be used to generate both summaries and keywords.

Appendices

Appendix A.

50 pre-trained topics (*gensim LDA* English model)

1. county, new, york, ohio, michigan, state, pennsylvania, township, florida, massachusetts, minnesota, virginia, city, creek, indiana, kentucky, iowa, jersey, boston, philadelphia
2. party, election, vote, minister, candidate, percentage, liberal, general, president, democratic, political, prime, member, elect, parliament, government, leader, new, seat, conservative
3. china, chinese, south, africa, iran, republic, pakistan, asia, korea, ali, african, kong, hong, islamic, thai, state, philippines, sco, arab, east
4. government, law, state, act, issue, public, service, policy, right, year, court, country, national, member, power, provide, report, plan, order, support
5. church, catholic, bishop, christian, pope, roman, unionist, saint, holy, cathedral, god, religious, mary, priest, canon, john, religion, orthodox, rome, faith
6. specie, plant, mountain, animal, bird, large, small, forest, white, tree, family, genus, dam, red, black, range, long, common, natural, wild
7. french, france, ndash, paris, louis, jean, saint, painter, les, charles, pierre, marie, paul, jacques, joseph, michel, victor, albert, henri, olivier

8. engine, car, model, design, vehicle, power, speed, drive, ford, version, build, new, series, produce, production, rear, year, motor, system, body
9. river, island, lake, park, north, south, water, bay, point, sea, coast, land, area, national, west, valley, beach, east, port, near
10. spanish, san, italian, mexico, spain, del, italy, puerto, santa, brazil, portugal, city, portuguese, rico, juan, mexican, antonio, maria, rio, argentina
11. century, history, empire, period, early, ancient, king, greek, roman, temple, modern, time, land, rule, territory, great, middle, date, kingdom, culture
12. day, man, time, go, little, white, night, good, leave, hand, old, tell, head, wear, woman, big, love, look, away, long
13. station, line, route, railway, road, highway, south, bus, bridge, street, north, service, australia, new, train, exit, australian, london, rail, east
14. john, david, michael, james, smith, paul, tom, jones, peter, robert, brown, jack, george, martin, harry, richard, frank, lee, jim, bill
15. work, social, time, group, view, believe, consider, question, talk, movement, idea, claim, term, human, describe, good, world, fact, write, point
16. california, san, washington, los, angeles, gold, francisco, city, seattle, center, arizona, colorado, american, nevada, oregon, silver, state, diego, winners, pacific
17. league, baseball, major, run, mdash, home, hit, red, chicago, career, hall, leader, series, bat, fame, detroit, york, field, plate, base
18. german, germany, von, van, wheel, grand, der, dutch, berlin, badge, prince, netherlands, duke, swiss, rover, die, austria, napoleon, austrian, und
19. house, building, build, park, site, area, open, old, street, centre, large, home, west, hill, castle, year, place, city, local, main
20. cite, web, software, system, user, support, information, title, version, access, file, card,

datum, journal, video, internet, site, release, url, computer

21. texas, carolina, tennessee, north, alabama, illinois, georgia, wisconsin, missouri, kansas, south, oklahoma, storm, mississippi, state, arkansas, concurrently, louisiana, lincoln, southern
22. cell, disease, medical, human, treatment, drug, cause, acid, health, blood, gene, body, syndrome, protein, study, brain, risk, medicine, bone, muscle
23. book, publish, write, work, new, novel, story, press, magazine, york, volume, world, author, life, writer, edition, marvel, vol, fiction, issue
24. university, school, college, education, year, program, research, institute, national, study, high, society, campus, center, member, work, hospital, academy, graduate, science
25. year, son, family, die, father, death, child, later, bear, return, time, wife, marry, brother, daughter, mother, life, young, leave, age
26. article, page, delete, wikipedia, category, shell, deletion, edit, comment, jewish, user, link, vote, think, don, discussion, israel, live, notable, jews
27. system, number, example, set, code, time, line, standard, value, case, model, function, form, base, order, point, object, require, ibm, key
28. radio, news, station, network, channel, television, new, broadcast, september, day, time, program, august, june, hour, year, host, bbc, january, october
29. canada, canadian, ontario, theory, quantum, toronto, group, field, representation, nova, matrix, citation, physics, begin, function, equation, space, provincial, scotia, theorem
30. united, states, judge, american, succession, state, serve, president, office, house, governor, senate, court, post, washington, congress, national, general, republican, war
31. high, low, process, energy, effect, material, water, result, increase, surface, cause, system, gas, rate, pressure, level, light, time, reduce, large
32. award, american, theatre, festival, actor, actress, opera, dance, writer, best, singer,

- stage, awards, politician, artist, director, player, bear, theater, film
33. language, india, indian, english, word, tribe, speak, form, letter, latin, grammar, dialect, verb, state, mumbai, native, dictionary, speaker, old, script
34. color, bar, text, till, black, width, value, end, shift, legend, start, blue, wikitable, align, layer, leave, gray, drum, style, class
35. town, city, district, area, row, village, population, region, locate, province, municipality, school, community, local, high, japan, merge, total, border, official
36. air, aircraft, flight, space, airport, force, mission, bmw, fly, launch, base, squadron, raf, crew, wing, pilot, nuclear, aviation, international, fighter
37. company, sell, product, market, business, new, bank, price, cylinder, production, group, store, produce, industry, oil, corporation, base, brand, acquire, sale
38. prefecture, moon, earth, cricket, food, ball, sun, test, spacecraft, solar, rotation, rice, day, milk, coin, orbit, wine, star, serve, tea
39. album, music, band, release, song, record, guitar, rock, single, live, new, play, tour, vocal, records, track, perform, feature, group, label
40. williams, walker, wright, clark, fork, rogers, spencer, vermont, amanda, gibson, watson, edwards, pyramid, dylan, covenant, stark, cobra, buffy, morton, shapiro
41. jpg, art, file, museum, image, new, work, design, painting, zealand, gallery, paint, style, artist, stamp, collection, architecture, sculpture, national, create
42. character, series, game, appear, man, comics, batman, power, kill, fantasy, fight, comic, version, release, earth, dark, reveal, universe, voice, story
43. russian, soviet, union, russia, war, polish, european, republic, poland, europe, germany, warsaw, hungary, german, world, moscow, swedish, sweden, hungarian, finland
44. british, john, london, william, england, george, royal, henry, sir, kingdom, charles, english, james, thomas, king, edward, scottish, queen, britain, scotland

-
45. team, football, league, club, cup, win, world, championship, national, match, final, stadium, year, sport, race, time, competition, chevrolet, title, division
 46. ireland, irish, quebec, des, northern, ulster, rose, dublin, hughes, lily, patrick, dame, sinn, moody, mac, lawrence, serge, sur, county, concerto
 47. war, army, force, battle, ship, military, division, navy, command, corps, regiment, attack, general, unit, service, infantry, operation, officer, naval, world
 48. film, series, episode, star, television, role, appear, play, movie, feature, character, direct, season, comedy, production, fox, drama, show, actor, release
 49. game, season, player, play, team, rugby, win, year, score, time, point, second, new, coach, record, round, start, end, hockey, games
 50. jul, aug, police, japan, kill, japanese, murder, crime, case, fire, prison, arrest, jun, death, trial, sentence, shoot, criminal, report, victim

Appendix B.

50 pre-trained topics (*sklearn LDA* English model)

1. group, plant, cell, food, animal, small, large, oil, form, female, product, male, common, produce, store, size, material, contain, grow, shape, variety, body, wear, human, natural, similar, base, consist, long, leave
2. ship, navy, uss, class, naval, sea, gun, fire, fleet, port, coast, crew, captain, service, operation, commission, command, royal, pacific, launch, arrive, officer, return, commander, bay, build, squadron, serve, carry, name
3. engine, car, model, design, vehicle, power, version, new, production, speed, drive, produce, ford, build, series, year, available, feature, company, market, introduce, sell, generation, standard, seat, replace, electric, high, base, light
4. company, court, law, act, business, service, government, state, year, office, sell, public, new, federal, judge, pay, market, case, purchase, price, financial, legal, acquire, provide, own, sale, right, justice, issue, private
5. hospital, department, medical, health, center, smith, memorial, care, general, city, doctor, university, service, centre, street, private, district, community, national, foundation, historical, facility, saint, heart, drug, public, practice, institute, day, research
6. air, aircraft, flight, force, airport, fly, space, squadron, mission, pilot, wing, base, launch, international, crew, service, operation, test, board, operate, training, passen-

- ger, ground, transport, world, land, station, kill, hour, military
7. album, release, song, record, track, single, records, music, feature, label, tour, live, studio, group, band, new, version, video, recording, cover, number, hit, write, title, debut, love, year, producer, original, produce
 8. century, early, history, period, time, year, old, ancient, modern, temple, great, late, roman, later, date, remain, culture, place, tradition, stone, form, middle, today, important, historical, empire, greek, day, accord, age
 9. radio, bank, green, branch, national, central, reserve, corporation, regional, rate, commercial, credit, hit, gold, account, network, today, fund, cross, federal, money, merge, tower, big, operation, red, establish, base, southern, large
 10. party, election, governor, vote, member, elect, candidate, general, committee, democratic, parliament, house, national, political, state, government, serve, senate, liberal, assembly, chairman, president, congress, leader, seat, council, office, secretary, win, new
 11. minister, president, prime, republic, state, council, head, monarch, government, general, premier, kingdom, secretary, leader, chief, king, sir, islands, queen, china, united, soviet, spain, federal, george, canada, high, germany, central, executive
 12. california, san, los, angeles, massachusetts, chicago, city, francisco, boston, johnson, american, mexico, pacific, york, state, center, valley, bay, southern, beach, washington, coast, america, mission, john, move, louis, begin, william, west
 13. specie, spanish, scottish, bird, family, scotland, spain, mexico, long, native, order, bill, small, head, short, large, america, common, brown, wing, black, number, world, live, member, forest, northern, ground, red, describe
 14. japan, japanese, china, chinese, ball, person, india, official, australia, pacific, total, expand, merge, march, october, create, medium, strike, corporation, heavy, hit, mountain, wing, drop, base, foot, main, family, forest, fan
 15. french, german, france, germany, italian, paris, von, italy, european, polish, louis, jean, europe, saint, charles, spain, russian, world, russia, paul, politician, die, war, joseph,

grand, jewish, foreign, bear, robert, august

16. british, london, royal, england, ireland, kingdom, irish, britain, united, english, van, queen, great, northern, society, sir, history, centre, empire, act, west, king, republic, european, appoint, union, theatre, title, europe, form
17. city, town, district, area, population, village, local, region, locate, municipality, school, centre, community, province, council, large, government, home, church, high, regional, resident, river, capital, primary, total, website, official, merge, small
18. time, end, leave, day, go, man, return, new, begin, later, place, year, run, start, set, follow, turn, long, good, try, attempt, lead, face, tell, head, fall, break, second, point, hand
19. high, cause, water, low, increase, effect, level, result, rate, energy, surface, test, occur, system, light, pressure, study, reduce, drug, condition, large, develop, blood, damage, require, area, control, process, report, lead
20. band, guitar, rock, vocal, bass, drum, metal, flag, lead, paul, play, engineer, member, electric, heavy, form, john, join, tom, death, black, bill, issue, magazine, replace, additional, design, influence, gold, dead
21. station, television, channel, network, news, bbc, broadcast, host, air, program, platform, digital, format, launch, hour, service, local, time, new, medium, feature, begin, available, show, live, announce, online, studio, website, own
22. music, play, festival, perform, dance, musical, performance, stage, singer, song, artist, tour, sound, work, live, new, love, voice, night, art, hall, popular, style, big, act, piece, time, major, recording, early
23. page, article, delete, wikipedia, link, edit, user, talk, think, don, add, vote, look, entry, good, live, place, notable, write, propose, search, information, issue, thing, create, site, agree, editor, fact, try
24. station, line, railway, road, street, route, highway, service, bridge, train, bus, open, rail, north, run, east, city, track, west, operate, build, cross, south, pass, new, serve, transport, section, connect, travel

25. language, english, word, speak, god, jewish, letter, write, greek, form, text, term, name, common, version, example, person, mark, follow, subject, case, accord, standard, native, official, read, origin, group, old, modern
26. united, states, american, washington, america, president, national, dog, post, congress, serve, canada, war, mexico, white, association, african, committee, kingdom, nation, country, north, native, city, international, organization, union, center, history, practice
27. new, york, book, publish, work, write, press, american, author, magazine, writer, university, collection, john, edition, library, story, history, art, editor, times, vol, paper, newspaper, review, novel, volume, robert, david, world
28. new, south, north, west, east, western, zealand, africa, southern, african, northern, eastern, central, region, coast, border, territory, pacific, state, indian, country, islands, union, area, regional, form, middle, white, governor, bay
29. category, shell, zealand, famous, file, figure, fight, field, festival, female, feel, federal, february, feature, father, fan, fame, family, final, fall, fail, fact, facility, face, eye, extend, express, experience, expand, exist
30. university, school, college, education, institute, campus, program, research, high, year, state, graduate, degree, national, professor, center, member, study, academy, hall, association, president, arts, board, offer, serve, public, institution, department, director
31. island, park, lake, area, mountain, national, land, bay, forest, valley, water, beach, hill, mount, point, north, river, near, islands, locate, sea, fort, site, range, large, south, rock, high, tree, west
32. game, player, season, league, play, baseball, team, major, hockey, run, basketball, series, career, home, year, record, coach, star, ice, new, hit, win, point, american, field, hall, leader, score, fame, time
33. war, battle, force, attack, army, world, military, fight, kill, man, capture, soldier, order, lead, weapon, campaign, civil, general, defeat, send, action, control, end, command, arm, power, later, second, support, victory
34. september, july, october, march, june, april, august, january, november, december,

february, day, ndash, year, month, announce, begin, new, later, report, return, week, second, follow, date, complete, end, late, name, move

35. bar, color, black, text, till, white, red, blue, brown, value, start, tree, end, leave, green, right, dark, line, year, non, mark, access, member, season, man, center, key, follow, death, woman
36. river, canada, canadian, creek, row, ontario, columbia, little, run, flow, north, federal, join, branch, lake, big, upper, west, note, province, rise, spring, district, south, name, indian, enter, fall, white, middle
37. social, group, country, movement, state, international, world, work, policy, government, organization, political, economic, human, community, society, member, right, national, development, public, support, individual, research, study, activity, issue, foreign, life, anti
38. army, soviet, military, russian, division, officer, union, general, corps, unit, regiment, service, police, russia, chief, force, rank, command, operation, camp, training, commander, staff, order, special, war, major, field, serve, defense
39. cite, web, book, title, journal, card, date, minor, planet, page, volume, news, work, author, center, discover, issue, name, access, year, earth, main, small, body, display, star, near, june, location, object
40. system, code, jul, program, software, technology, support, control, user, device, information, machine, datum, project, provide, base, design, key, application, development, access, video, source, allow, develop, standard, process, network, service, management
41. number, example, point, theory, set, function, form, field, value, method, case, order, space, time, term, define, object, problem, line, follow, structure, element, result, model, require, base, apply, system, end, non
42. team, football, league, cup, club, win, championship, season, national, world, match, play, final, stadium, year, rugby, division, tournament, player, score, goal, sport, round, second, champion, finish, defeat, title, professional, coach
43. john, william, australia, james, george, australian, victoria, thomas, wales, henry, sir,

robert, charles, david, edward, earl, richard, melbourne, lord, jones, williams, michael, english, politician, peter, paul, england, smith, frank, martin

44. county, state, pennsylvania, texas, ohio, township, michigan, virginia, carolina, florida, illinois, minnesota, georgia, jersey, city, york, lake, north, fort, washington, historic, valley, new, community, district, west, spring, lee, louis, grand

45. church, king, prince, saint, duke, catholic, kingdom, christian, bishop, roman, emperor, empire, castle, mary, joseph, grand, count, charles, john, peter, queen, alexander, william, order, great, title, henry, george, royal, paul

46. child, die, year, family, son, father, bear, death, life, work, woman, young, later, marry, wife, age, daughter, brother, mother, live, man, house, time, return, old, friend, murder, name, leave, home

47. award, world, win, race, year, medal, gold, prize, grand, best, academy, event, international, summer, good, horse, winner, receive, national, time, hold, competition, star, record, second, run, annual, honor, career, bear

48. film, episode, star, series, television, role, actor, movie, play, theatre, appear, character, feature, season, direct, director, production, american, cast, release, producer, scene, video, man, produce, story, title, base, company, jack

49. character, series, india, story, indian, novel, earth, appear, power, book, kill, version, human, world, create, dark, super, doctor, issue, dead, original, reveal, voice, planet, boy, publish, master, black, new, life

50. jpg, file, building, museum, house, design, art, build, image, tower, architecture, site, open, square, hall, construction, wall, garden, new, room, style, work, large, historic, complete, office, national, city, main, home

Appendix C.

50 pre-trained topics (*gensim LDA* German model)

1. band, album, jazz, song, hit, woche, spielen, lied, musik, single, music, live, rock, musiker, erscheinen, charteintrag, titel, records, love, one
2. katholisch, papst, erzbischof, bischof, bistum, erzbistum, kirche, johannes, katholische, paul, jerusalem, santa, rom, kardinal, heilige, mailand, diplomat, droge, schuppen, georgien
3. staatsanwaltschaft, soldat, weltkrieg, krieg, boot, armee, angriff, waffe, japanisch, japan, klasse, dienst, rebell, patient, oktober, republikaner, einheit, deutsch, november, april
4. system, verwenden, entwickeln, bereich, donnerstag, hersteller, bestehen, hoch, finden, unterschiedlich, zeigen, form, zahl, bestimmen, kunde, forschler, entwicklung, gering, stark, beispielsweise
5. russland, syrien, russisch, moskau, ungarn, russische, diensttag, alexander, irak, sowjetisch, sowjetunion, botschafter, zelle, sozialdemokrat, parteichef, dublin, petersburg, sankt, atv, oblast
6. university, open, press, london, college, school, american, brown, society, metal, schlagzeuger, history, saxophonist, bassist, academy, america, cambridge, international, lewis, homepage

7. krieg, iran, provinz, land, reich, hauptstadt, opposition, schlacht, hessen, armee, regierung, ministerium, serbien, kiew, fraktion, staat, islam, truppe, gebiet, region
8. griechenland, trump, athen, berg, wahlkampf, mexiko, donald, mexikanisch, augsburg, ingolstadt, mingus, bayerische, spital, alpe, gabriel, gen, adam, attacke, entspringen, bamberg
9. maria, joseph, franzen, bellen, josef, prinz, ferdinand, orden, regiment, jakob, rtl, albert, leo, priester, sierra, don, kaiser, brigade, prinzessin, messe
10. new, staat, york, vereinigte, city, state, amerikanisch, vereinigt, washington, north, bay, johnson, texas, san, amerikanische, south, angeles, boston, party, district
11. web, google, nutzer, datum, nummer, version, internet, software, aktuell, programm, video, zeile, plattform, information, entwickler, game, start, update, system, inhalt
12. peter, hans, deutsch, christian, werner, heinz, michael, andrea, wolfgang, walter, fischer, thoma, martin, ernst, klaus, paul, anna, rudolf, joachim, gerhard
13. gewinnen, platz, team, sieg, saison, punkt, spiel, weltmeisterschaft, cup, meisterschaft, finale, rund, turnier, erreichen, minute, europameisterschaft, olympisch, erfolg, titel, olympische
14. museum, graz, kunst, linz, stadt, ausstellung, architekt, werk, maler, sammlung, kultur, arbeit, art, haus, galerie, villa, architektur, bildhauer, schaffen, malerei
15. spiel, spielen, spieler, sport, saison, league, liga, verein, trainer, mannschaft, bundesliga, tor, wechseln, stadion, vertrag, partie, klub, nationalmannschaft, club, erzielen
16. bild, karte, foto, kamera, post, lizenz, bull, logo, klagenfurt, fotograf, level, pixel, israelische, fotografieren, verbraucher, objekt, aufnahme, grafik, domain, badminton
17. film, frau, kind, leben, mutter, freund, familie, vater, spielen, eltern, drehen, haus, tochter, freundin, lernen, alt, fliehen, sterben, arbeiten, drehbuch
18. kategorie, international, festival, internationale, ungarisch, beste, bester, preis, oscar, nation, nominierung, jahr, vereinte, awards, ungarische, vienna, nominieren, auszeich-

nen, year, navigationsleiste

19. sprache, van, griechisch, griechische, arabisch, tirol, tiroler, angreifer, bezeichnen, pferd, name, griechen, bauer, jahrhundert, zeichen, grab, quelle, mittelalter, alphabet, schrift
20. china, regie, theater, chinesisches, salzburger, indien, australien, schauspieler, tatort, rolle, kongo, regisseur, indisch, deal, afghanistan, asien, autonom, kino, spielen, sat
21. wien, wiener, verlag, austria, zeitung, literatur, titel, akademie, seite, erscheinen, ausgabe, autor, steiermark, datum, vorarlberg, burgenland, schelling, zeitschrift, ort, band
22. jpg, salzburg, burg, schloss, haus, jahrhundert, datei, stein, kreuz, besitz, autobahn, familie, mauer, anlage, wald, tor, novelle, garten, linzer, konrad
23. deutsch, person, kanton, familienname, politiker, schweizer, amerikanisch, name, etappe, route, tour, schriftsteller, schauspieler, jurist, causa, komponist, maler, journalist, monument, continental
24. paris, frankreich, jean, saint, belgien, herzogtum, france, herrscher, pierre, luxemburg, pariser, louis, franzose, niederlande, libyen, hollande, marie, michel, italien, belgisch
25. bahnhof, strecke, linie, bahn, bau, flughafen, betrieb, bauen, stadt, zug, freitag, neu, errichten, richtung, nutzen, kilometer, beginnen, west, dortmund, verkehr
26. mittwoch, android, bad, slowenien, gipfel, vorjahr, stunde, haushalt, via, bus, filiale, freitagabend, duell, slowenisch, link, xbox, sony, betragen, einkommen, links
27. standard, frage, international, wirtschaft, europa, thema, staat, deutschland, facebook, land, euro, derzeit, medium, usa, wichtig, sozial, entscheidung, ziel, problem, arbeit
28. syrisch, syrer, station, leichtathletik, finnisch, finnland, sport, weltcup, nsa, street, line, wirtschaftskammer, fbi, watch, abfahrt, min, alpin, drive, ski, helsinki
29. award, david, michael, john, amerikanisch, frank, schauspieler, staffel, james, paul, schauspielerin, jack, green, rolle, story, steve, joe, king, regisseur, hollywood

30. schule, kind, lehrer, arbeiten, schweiz, gymnasium, besuchen, basel, erhalten, oper, angela, junge, ausbildung, jugendliche, freiheitliche, unterrichten, klasse, leben, lernen, schweizer
31. berlin, deutschland, deutsch, deutschen, deutsche, hamburg, berliner, ddr, deutscher, bremen, sachsen, dresden, hannover, weltkrieg, brandenburg, cdu, hamburger, holstein, anhalt, bundesrepublik
32. leben, buch, geschichte, werk, roman, lieben, welt, wort, erscheinen, kultur, jahrhundert, religion, autor, schriftsteller, text, gedicht, hypo, titel, tod, jude
33. schwede, schiff, schwedisch, rheinland, pfalz, bord, schiffe, ozean, mittelmeer, stern, bmw, krefeld, rhein, franke, meer, roma, fuchs, beach, flotte, becken
34. prozent, euro, unternehmen, million, milliarde, konzern, firma, wirtschaft, spanisch, mitarbeiter, gmbh, verkaufen, spanien, mio, steigen, markt, weltweit, deutschland, aktie, brasilien
35. diensttag, apple, windows, microsoft, online, media, amazon, server, justiz, betriebssystem, rechner, kandidieren, dienst, bowl, volkswagen, zugriff, web, basketballnationalmannschaft, speed, verschicken
36. gemeinde, stadt, ort, kilometer, einwohner, region, jahrhundert, gebiet, dorf, landkreis, name, kreis, norden, westen, osten, ortsteil, provinz, siedlung, bezirk, westlich
37. union, ausschuss, rio, olympia, grand, rakete, england, rugby, irland, wal, match, flagge, las, camp, edge, schottland, slam, style, class, janeiro
38. sohn, familie, tod, sterben, vater, ehe, tochter, kind, bruder, alt, festnehmen, heiraten, gebären, frau, haus, erhalten, leben, peking, schloss, mutter
39. freitag, serie, welt, fan, nachrichtenagentur, erscheinen, magazin, folge, figur, publikum, show, star, comic, news, reihe, lieben, geschichte, arbeiten, auftritt, name
40. frankfurt, river, main, ermittler, mainz, frankfurter, krankenhaus, abteilung, volltext, insekt, psychologie, wiesbaden, weibchen, creek, ei, howard, soziologie, zelte, books, jochen

-
41. finden, laut, zeigen, panorama, polizei, berichten, woche, montag, schwer, kommen, monat, sonntag, halten, setzen, mann, versuchen, fast, treffen, leben, geld
 42. kirche, jahrhundert, kloster, errichten, evangelisch, erhalten, turm, stammen, kapelle, bau, syrische, heilig, alt, pfarrer, miliz, finden, erbauen, heilige, neu, evangelische
 43. sender, weiterleitung, sendung, radio, konzert, hotel, apps, fernsehen, rundfunk, bbc, gesang, tiger, programm, johnny, schlagzeug, model, live, malaysia, wdr, dan
 44. italienisch, italien, san, del, wasser, italienische, rom, temperatur, antonio, gas, chemisch, giovanni, erkrankung, updates, slowakei, verbindung, nash, kraftwerk, fabrik, mayr
 45. auto, fahrzeug, bank, motor, modell, mark, wagen, county, liter, pkw, cook, baureihe, audi, porsche, valley, hongkong, daimler, historic, shanghai, benz
 46. institut, wissenschaft, leipzig, professor, mitglied, studieren, berlin, stuttgart, deutsch, innsbruck, gesellschaft, geschichte, hochschule, studium, bonn, freiburg, auflage, wissenschaftlich, arbeit, band
 47. rennen, preis, premier, fahrer, formel, fahren, sport, football, explosion, runde, mercedes, chemie, speedway, planet, lkw, ceo, prix, stunde, vergeben, physik
 48. john, london, britisch, william, robert, george, englisch, charles, donnerstag, britische, nascar, kroatien, chicago, richard, pic, thoma, bill, james, serbisch, henry
 49. benutzer, wikipedia, artikel, diskussion, seite, frage, problem, dank, hab, sperren, hallo, finden, fall, sinn, genau, diskussionsseite, urv, version, falsch, link
 50. partei, regierung, wahl, mitglied, kandidat, parlament, juni, politiker, juli, november, september, oktober, abgeordnete, stimme, amt, international, prozent, politisch, april, februar

Appendix D.

50 pre-trained topics (*sklearn LDA* German model)

1. berlin, leipzig, ddr, berliner, schweiz, schweizer, dresden, theater, brandenburg, halle, sachsen, deutschen, kunst, deutschland, hochschule, max, freie, otto, carl, vertreten
2. art, band, familie, gattung, leben, adam, verbreiten, county, ordnung, gruppe, vertreter, innerhalb, europa, bilden, wachsen, eintrag, einzig, benennen, form, lexikon
3. graf, burg, schloss, grafenschaft, herzog, herzogtum, kaiser, herr, heinrich, rhein, besitz, herrschaft, ulrich, bischof, sohn, linie, mark, tochter, verkaufen, kloster
4. mitglied, institut, studieren, arbeiten, gesellschaft, professor, akademie, studium, hochschule, arbeit, wissenschaft, erhalten, leiter, philosophie, geschichte, direktor, freiburg, vorsitzender, besuchen, treten
5. kreis, landkreis, ort, berg, karte, baden, ortsteil, bayer, gemeinde, bezirk, stadt, hessen, westfale, amt, wald, sachsen, land, bad, sitz, bezeichnen
6. unternehmen, prozent, million, euro, gmbh, mark, firma, wirtschaft, dollar, mitarbeiter, geld, verkaufen, markt, produkt, kosten, sitz, produktion, weltweit, deutschland, anteil
7. wien, peter, michael, wiener, thoma, martin, andrea, christian, regie, alexander, daniel, stefan, wolfgang, frank, franzen, heinz, homepage, christoph, josef, ernst

8. insel, tier, world, see, wasser, island, west, nord, pflanze, hotel, boden, kopf, entdecken, westlich, benennen, gelegen, feld, trennen, russland, besuchen
9. modell, motor, fahrzeug, rennen, auto, einsatz, wagen, typ, version, bauen, einsetzen, hersteller, fahren, division, entwickeln, variante, leistung, erhalten, firma, herstellen
10. deutschland, deutsch, deutschen, deutsche, deutscher, bund, ausbildung, polizei, verband, dienst, mainz, mitglied, organisation, jugend, rahmen, absolvieren, vertreten, setzen, herbst, gold
11. partei, regierung, politisch, krieg, armee, wahl, russisch, mitglied, politische, soldat, schlacht, parlament, general, russland, gesetz, amt, truppe, staat, organisation, weltkrieg
12. film, frau, kind, leben, mann, vater, award, tod, mutter, spielen, sterben, familie, sohn, lieben, tochter, alt, rolle, freund, regisseur, schauspieler
13. meter, finden, schwarz, hoch, rot, farbe, breit, stark, zeigen, selten, besitzen, bilden, blau, klein, erreichen, lang, seite, ober, link, fast
14. friedrich, johann, karl, wilhelm, heinrich, ludwig, georg, haus, franzen, otto, carl, ernst, august, sohn, hermann, herrscher, philipp, anton, adolf, ferdinand
15. new, york, museum, university, london, city, kunst, press, ausstellung, college, school, sammlung, werk, national, maler, american, arbeiten, modern, center, studieren
16. land, china, europa, welt, flughafen, chinesisches, kultur, republik, international, grenze, hauptstadt, stein, zentrum, reise, region, osten, nahe, leben, national, wirtschaft
17. entwicklung, aufgabe, arbeit, untersuchung, rolle, studie, thema, sozial, wichtig, bereich, entwickeln, forschung, begriff, bewegung, ziel, theorie, spielen, wissenschaftlich, ergebnis, historisch
18. benutzer, artikel, relevanz, behalten, relevant, wikipedia, lemma, finden, diskussion, begriff, thek, quelle, fall, liste, eher, thema, codeispoetry, kungfuman, inhalt, google
19. album, band, musik, hit, lied, jazz, spielen, single, song, pianist, live, musiker, music,

-
- saxophonist, rock, komponist, erscheinen, konzert, festival, sendung
20. frage, finden, problem, fall, zeigen, person, genau, gelten, verstehen, handeln, wissen, eher, glauben, bekommen, klaren, scheinen, thema, laut, halten, sogar
21. bischof, katholisch, kirche, bistum, maria, salzburg, johannes, erzbischof, papst, hochstift, katholische, ungary, paul, joseph, heilige, josef, rom, ernennen, rat, amt
22. bild, schule, wappen, kloster, bad, golden, darstellung, lehrer, zeigen, darstellen, feld, kind, freie, figur, rot, belegen, aufnahme, alt, haus, kunst
23. benutzer, wikipedia, artikel, diskussion, seite, dank, sperren, hallo, version, sinn, complex, diskussionsseite, bild, stunde, link, marcus, felix, links, rainer, stefan
24. spiel, saison, spieler, spielen, team, wechseln, cup, league, eishockey, trainer, nationalmannschaft, weltmeisterschaft, erzielen, karriere, mannschaft, liga, gewinnen, vertrag, hockey, tor
25. liste, weiterleitung, griechisch, griechenland, reich, geld, china, anzahl, deutsche, robert, normenliste, psychotria, center, gattung, gelten, foto, frage, gelingen, gelegen, fragen
26. punkt, erreichen, gelingen, verlieren, minute, seite, letzt, setzen, ziehen, erneut, treffen, schwer, sieg, sichern, knapp, gewinnen, angriff, beginnen, fallen, folgen
27. woche, datum, laut, online, zeitung, web, seite, angabe, information, berichten, monat, panorama, internet, software, foto, montag, google, medium, aktuell, bericht
28. name, england, opfer, niederlande, gericht, zahl, benennen, reich, fort, bedeuten, bezeichnung, erinnern, bekannt, leiten, entdecken, fallen, bekommen, ehemalig, stammen, annehmen
29. international, preis, internationale, union, japan, website, hannover, offiziell, club, wetbewerb, braunschweig, finden, japanisch, junge, jeweils, rahmen, stattfinden, vertreten, jed, ehemalig
30. hamburg, frankfurt, main, gymnasium, stiftung, stuttgart, bremen, orden, archiv, albert, alte, deutsch, geschichte, schule, kunst, selben, bonn, wilhelm, frau, april

31. verlag, hans, literatur, roman, auflage, walter, geschichte, autor, werner, wolfgang, max, deutsch, fischer, stuttgart, fritz, heinz, bernhard, landtag, bundesrepublik, buch
32. neu, projekt, erolgen, finden, stunde, bieten, bestehen, nutzen, bau, anlage, bereich, entstehen, beginnen, rahmen, planen, veranstaltung, betrieb, einrichtung, direkt, erhalten
33. paris, frankreich, italienisch, italien, jean, louis, saint, rom, marie, spanien, simon, dame, charles, paul, ungar, joseph, niederlande, griechenland, national, sohn
34. bahnhof, strecke, linie, bahn, bahnstrecke, betrieb, baureihe, zug, richtung, abschnitt, station, kilometer, verbindung, fahren, bau, gesellschaft, beginnen, rhein, bauen, nord
35. kategorie, san, klasse, del, boot, schiff, com, portal, royal, eintrag, laufen, weltkrieg, sammeln, spanien, gesamt, mehrfach, weit, psychotria, teil, frau
36. deutsch, person, familienname, van, politiker, amerikanisch, maler, schriftsteller, schauspieler, schauspielerin, komponist, journalist, architekt, schweizer, amerikanische, autor, musiker, paul, trainer, ehemalig
37. juli, august, oktober, juni, september, april, november, januar, dezember, februar, jahr, dienst, monat, erolgen, anfang, treten, finden, erhalten, ehemalig, stand
38. verein, bundesliga, mannschaft, liga, tor, sport, spielen, saison, pokal, meisterschaft, platz, jugend, steigen, deutsch, verband, bayer, erfolg, finale, frau, erreichen
39. sprache, englisch, spanisch, bezeichnen, wort, svg, begriff, verwenden, englische, schrift, bezeichnung, zitat, spanien, don, stammen, lied, form, deutschsprachig, bedeuten, benutzen
40. gemeinde, kanton, provinz, dorf, kilometer, ort, region, einwohner, norden, gebiet, westen, saint, tal, ortschaft, osten, bestehen, frankreich, westlich, grenze, leben
41. staat, amerikanisch, usa, vereinigte, vereinigt, national, united, amerikanische, american, kanada, washington, university, dollar, center, ort, thoma, betragen, state, organisation, fort

-
42. text, werk, quelle, bedeutung, satz, stil, gitarre, bedeutend, autor, top, politisch, start, teil, stammen, literatur, enthalten, angeben, hinaus, abschnitt, version
 43. jpg, kirche, evangelisch, datei, turm, kreuz, kapelle, gemeinde, erbauen, errichten, blick, stammen, architekt, bau, lage, erhalten, heilige, schaffen, ersetzen, martin
 44. bar, system, verwenden, form, bestehen, bezeichnen, bestimmen, unterschiedlich, gering, beispielsweise, regel, einsetzen, einzeln, entsprechen, hoch, verfahren, verbindung, funktion, entwickeln, bilden
 45. standard, titel, eins, zeile, ausgabe, erscheinen, buch, magazin, video, zeitschrift, einheit, thema, autor, richten, interessieren, international, auftrag, deutschsprachig, bekannt, sammlung
 46. serie, folge, staffel, figur, geschichte, formel, japanisch, reihe, erscheinen, zeichnen, produzieren, autor, amerikanisch, schaffen, divers, fan, handeln, lauf, gestalten, sender
 47. platz, gewinnen, weltmeisterschaft, olympisch, schwedisch, open, olympische, schwede, turnier, europameisterschaft, meisterschaft, sieg, grand, spiel, rang, finale, moskau, belegen, international, rennen
 48. john, william, george, robert, britisch, david, james, park, till, richard, river, charles, state, henry, north, frank, london, paul, king, royal
 49. stadt, jahrhundert, erhalten, errichten, alt, weltkrieg, entstehen, ehemalg, polnisch, krieg, stammen, finden, ort, mitte, erbauen, reich, historisch, siedlung, gebiet, bau
 50. gruppe, nummer, programm, air, tour, etappe, sender, radio, bank, start, fernsehen, team, starten, center, ziel, entscheiden, fahren, ehemalg, beenden, positiv

Appendix E.

Generated Summaries

E.1. English gensim Summary

E.1.1. Without considering a position structure

I do not see any other way to determine whether one is able to be successful at computer science studies. Then students, who are not ready to study in the field of computer science, can not start the studies. One can determine the suitability for the study generally only during the study. You could design the aptitude test such that students can still start the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses.

E.1.2. Considering a position structure

I do not see any other way to determine whether one is able to be successful at computer science studies. Then students, who are not ready to study in the field of computer science, can not start the studies. One can determine the suitability for the study generally only during the study. You could design the aptitude test such that students can still start the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer

science courses.

E.2. English TextRank Summary

E.2.1. Without considering a position structure

As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. Then students, who are not ready to study in the field of computer science, can not start the studies. I do not see any other way to determine whether one is able to be successful at computer science studies. Studying not only imparts knowledge, but also key competencies, which may be important in other disciplines. One takes valuable time from the students by this.

E.2.2. Considering a position structure

As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. Then students, who are not ready to study in the field of computer science, can not start the studies. I do not see any other way to determine whether one is able to be successful at computer science studies. Studying not only imparts knowledge, but also key competencies, which may be important in other disciplines. One takes valuable time from the students by this.

E.3. English TF Summary

E.3.1. Without considering a position structure

I do not see any other way to determine whether one is able to be successful at computer science studies. Then students, who are not ready to study in the field of computer science, can not start the studies. One can determine the suitability for the study generally only during the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. You could design the aptitude test such that students can still start the study. I basically do not see the need for an aptitude test

E.3.2. Considering a position structure

Then students, who are not ready to study in the field of computer science, can not start the studies. One can determine the suitability for the study generally only during the study. I do not see any other way to determine whether one is able to be successful at computer science studies. You could design the aptitude test such that students can still start the study. As long as one is able to study computer science as a secondary subject of an admission-free subject, a suitability test will not be able to prevent anyone from visiting the computer science courses. One takes valuable time from the students by this. It would be helpful for students to get an assessment at the beginning.

E.4. English TF-IDF Summary

E.4.1. Without considering a position structure

This will be shown in the first semesters anyway. It would be helpful for students to get an assessment at the beginning. one takes valuable time from the students by this. I basically do not see the need for an aptitude test. One can determine the suitability for the study generally

only during the study. Studying not only imparts knowledge, but also key competencies, which may be important in other disciplines.

E.4.2. Considering a position structure

This will be shown in the first semesters anyway. One takes valuable time from the students by this. It would be helpful for students to get an assessment at the beginning. I basically do not see the need for an aptitude test. One can determine the suitability for the study generally only during the study. Studying not only imparts knowledge, but also key competencies, which may be important in other disciplines.

E.5. German gensem Summary

E.5.1. Without considering a position structure

Ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen. Solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. man damit den Studenten wertvolle Zeit wegnimmt. So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können.

E.5.2. Considering a position structure

Ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen. Solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann.

So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können.

E.6. German TextRank Summary

E.6.1. Without considering a position structure

Solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird. So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten man damit den Studenten wertvolle Zeit wegnimmt. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. Das Studium nicht nur Fachwissen, sondern auch Schlüsselkompetenzen vermittelt, die in anderen Disziplinen wichtig sein können.

E.6.2. Considering a position structure

Solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird. So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten man damit den Studenten wertvolle Zeit wegnimmt. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. Das Studium nicht nur Fachwissen, sondern auch Schlüsselkompetenzen vermittelt, die in anderen Disziplinen wichtig sein können.

E.7. German TF Summary

E.7.1. Considering a position structure

So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten. Solange man Informatik als Nebenfach eines zulassungsfreien Faches studieren kann, ein Eignungstest niemanden vor dem Informatik-Studium abhalten können wird ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen. Es für Studierende hilfreich wäre, vorher eine Einschätzung zu bekommen. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. ich grundsätzlich die Notwendigkeit einer Eignungsprüfung nicht sehe.

E.7.2. Considering a position structure

So Studenten, die nicht bereit sind, ein Studium im Fach Informatik anzufangen, dieß nicht anfangen können. Man damit den Studenten wertvolle Zeit wegnimmt. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. Ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen. Sich das ohnehin in den ersten Semester zeigt. Man kann den Eignungstest ja so gestalten, dass Studierende dennoch das Studium aufnehmen könnten. Das Studium nicht nur Fachwissen, sondern auch Schlüsselkompetenzen vermittelt, die in anderen Disziplinen wichtig sein können.

E.8. German TF-IDF Summary

E.8.1. Without considering a position structure

Es fast immer eine andere Möglichkeit/Lösung gibt. Sich das ohnehin in den ersten Semester zeigt. Man damit den Studenten wertvolle Zeit wegnimmt. Ich grundsätzlich die Notwendigkeit einer Eignungsprüfung nicht sehe es für Studierende hilfreich wäre, vorher eine Einschätzung zu bekommen. Ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium

festzustellen. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann.

E.8.2. Considering a position structure

Es fast immer eine andere Möglichkeit/Lösung gibt. Sich das ohnehin in den ersten Semester zeigt. Man damit den Studenten wertvolle Zeit wegnimmt. Ich grundsätzlich die Notwendigkeit einer Eignungsprüfung nicht sehe. Man kann die Eignung für das Studium allgemein erst während des Studiums fest stellen kann. Es für Studierende hilfreich wäre, vorher eine Einschätzung zu bekommen. Ich keine andere Möglichkeit sehe, die Eignung zum Informatikstudium festzustellen.

E.9. Reference Summaries

	English	German
1	<p>Some students proposed to create an aptitude test, in order to check if new students have all required competencies or not. Though, some students think that such a test won't solve the problem, because it is allowed to study the computer science as a secondary admission-free subject.</p>	<p>Es macht Sinn, dass die Studenten vor dem Beginn des Studiums einen Eignungstest machen. Dieser Test könnte hilfreich sein, um festzustellen, ob die Studenten genug Kompetenzen für das Studium besitzen. Obwohl da man Informatik als ein zulassungsfreies Fach studieren kann, wird so ein Test nicht viel bringen.</p>
2	<p>It would be helpful for students to take an assessment test at the beginning of the studies. This way, one can determine the suitability and self-motivation for the subject. Students who fails appear to not be suitable for computer science.</p>	<p>Der Text nennt Begründungen für eine nicht genannte These. Es werden Argumente dafür und dagegen genannt.</p>
3	<p>There are pros and cons for the creation of an aptitude test. On one side, it can determine the skills of students and prevent less skillful students from studying. On the other side, a lot of students drop out because of the lack of self-motivation. In this case a suitability test will not change anything.</p>	<p>Man sollte einen Eignungstest erstellen, der eine Einschätzung geben könnte, ob Studenten für das Informatik-Studium geeignet ist. Obwohl viele Studenten scheitern nicht weil sie nicht genug Kenntnisse haben, sondern weil ihnen die Selbstmotivation fehlt.</p>

Table E.1 continued from previous page

4	<p>Some participants expressed an opinion that a suitability test has to be created. This test could help to define the skills level of students. Ans some participants are against such an aptitude test, because this might prevent students who can study but lack self-motivation from studying.</p> <p>Besides, since computer science can be studied as a secondary subject, such test won't prevent unskilled students from studying.</p>	<p>Die Eignung für das Informatik-Studium lässt sich nicht für alle vor dem Studium feststellen, da ein Studium sich stark von den bisherigen Lernerfahrungen und -motivationen der Schule unterscheidet und sich für manche erst im Laufe des Studiums erschließt. Es ist hilfreich, vor Beginn des Studiums eine objektive Einschätzung der persönlichen Eignung für ein Informatik-Studium zu erhalten, welche niemanden von einem Informatik-Studium abhält.</p> <p>Durch den Aufwand einer Eignungsprüfung nimmt man den Studierenden wertvolle Zeit. Auch, wenn sie das Studium nicht erfolgreich zu Ende führen, lernen Sie interdisziplinär anwendbare Schlüsselkenntnisse kennen.</p> <p>Eine passende Lösung sollte auch fair sein, sodass sich niemand über Umwege oder Tricks einen Studienplatz sichern kann, obwohl ein gegebener Eignungstest diese Person von einem Studium ausgeschlossen hätte.</p>
---	--	---

Table E.1 continued from previous page

5	<p>Suitability for a course of study can only be evaluated while studying. Some people do not see a need for a test. On the other hand, a self-assessment at the beginning might be helpful.</p>	<p>Ein Eignungstest kann helfen, dass Studenten erkennen, ob sie für das Studium geeignet sind. Aber man den Eignungstest auch umgehen kann und sowieso erst im Lauf der ersten Semester feststellen kann, ob man geeignet ist oder nicht.</p>
6	<p>It would be helpful to create an aptitude test to get an assesment of students' skills at the beginning. But such a test may fail to determine if students are able to study computer science or not. The reason is because a lot of students fail because of lack of self-motivation, and this can only be seen during the study.</p>	<p>Es handelt sich um eine Liste von Satzfragmenten, die anscheinend Argumente für und gegen eine Eignungsprüfung in der Informatik geben sollen. Sie beinhaltet viele Rechtschreib- und Grammatikfehler.</p>
7	-	<p>Es gibt die beiden Positionen pro und contra Eignungstest im Informatikstudium. Auf der Contra-Seite wird die Meinung vertreten, dass Studenten während des Studiums überhaupt ihre Eignung für das Fach feststellen. Auf der Pro-Seite plädiert man dafür, dass ungeeignete Kandidaten von vornherein ausgesiebt werden.</p>

Table E.1.: Human-produced reference summaries gathered during the evaluation step.

Appendix F.

Generated Keywords

	English	German
gensim	studies studying, study, valuable, test, self, subject	ein, eine, eines, studium, studiums, nicht, das, sehe die, können man kann, informatik, studenten, den, studierende, ich, für, eignung, möglichkeit, andere, eignungstest, schule eher zweitrangig, fach, faches, disziplinen wichtig, zeit, semester
Rake	takes valuable time, computer science courses, computer science studies, study computer science, computer science, admission-free subject, imparts knowledge, key competencies, aptitude test, lack self-motivation, study generally, secondary subject	studiums fest stellen, schule eher zweitrangig, anderen disziplinen wichtig, zulassungsfreien faches studieren, fach informatik anzufangen, ersten semester zeigt, andere möglichkeit/lösung gibt, viele studenten scheitern, studierende hilfreich wäre, solange man informatik, studium aufnehmen könnten, so studenten, studierende dennoch
TF	computer, science, able, study, test, aptitude, secondary, subject, suitability, way, successfull, need, admission, free, course, helpful, student, assessment, beginning, semester, knowledge, key, competency, important, discipline, ready, field, valuable, time, self, motivation, school	Studium, Eignung, Informatik, Eignungstest, Studierende, Möglichkeit, Informatikstudium, allgemein, fest, Student, Selbstmotivation, Schule, zweitrangig, bereiten, Fach, dieß, wertvolle, Möglichkeit/Lösung, Fachwissen, Schlüsselkompetenzen, Disziplin, wichtig, Semester, Nebenfach, zulassungsfreien, Informatik-Studium, hilfreich, Einschätzung, grundsätzlich, Notwendigkeit, Eignungsprüfung
TF (improved)	computer, science, study, test, suitability, aptitude, subject, way, time, student, field, self, motivation, school, assessment, beginning, knowledge, competency, discipline, semester, need, admission, course	Eignung, Informatik, Studierende, Eignungstest, Möglichkeit, Informatikstudium, Studium, Fachwissen, Schlüsselkompetenzen, Disziplin, Student, Fach, dieß, Notwendigkeit, Eignungsprüfung, Einschätzung, Möglichkeit/Lösung, Nebenfach, Informatik-Studium, Selbstmotivation, Schule, Semester

Table F.1.: The keywords generated for the evaluation with the following parameters: positionID = 320 (English) and positionID = 50 (German), ratio = 0.5, considerStructure = false.

	English	German
1	Computer science studies, aptitude test, suitability test, key competencies	Informatik, Studium, Einschätzung, Schlüsselkompetenzen, Eignungstest
2	computer science, study, competency, assessment	Informatik, Schlüsselkompetenzen, Eignungstest
3	key competence, computer science, aptitude test, suitability, self-motivation	Informatik, Einschätzung, Selbstmotivation, Schlüsselkompetenzen, Eignungstest
4	Computer science, aptitude test, key competencies, suitability, self-motivation	Eignungsfeststellung, Informatikstudium, Diskussion
5	computer science, assessment, suitability test, knowledge	Informatik, Einschätzung, Selbstmotivation, Fachwissen, Test
6	computer science, key competencies, secondary subject, admission, suitability	Informatik, Studium, Eignung, Test, scheitern, Motivation, Zulassung, Einschätzung
7	computer science, key competencies, admission, suitability, aptitude test	Studium, Eignungstest, Eignung für Studium, Semester, Selbstmotivation, Notwendigkeit
8	computer science, suitability, aptitude test, assessment	Eignungstest, Informatik, pro und contra, Liste
9	computer science, suitability, assessment, aptitude test	Eignungstest, Informatik-Studium, Selbstmotivation, Schlüsselkompetenz
10	computer science, suitability, assessment	Eignungstest, Informatikstudium, Selbstmotivation, Einschätzung, Schlüsselkompetenzen
11	-	Informatik, Eignungstest, Einschätzung, Selbstmotivation, Eignung

Table F.2.: The reference keywords given by the participants of the evaluation.

List of Figures

3.1.	Example graph in <i>D-BAS</i> [Net20].	9
3.2.	Example of statements.	10
3.3.	Example of an argument.	10
3.4.	Example of a position (the statement in the blue box).	11
3.5.	Example of various attack types.	11
4.1.	An example text document. Related words of the same color can be grouped into one topic. Original image from [BNJ03].	16
4.2.	A possible association of documents to topics from Figure 4.1. For simplicity, only three topics were chosen.	17
4.3.	A visualization of a dataset as a matrix factorization problem [Suh+16]. . . .	18
4.4.	An example of a generative process for an LDA model. Graphic taken from: [BCD10].	19
4.5.	A plate notation of a pLSA model [BNJ03].	22
4.6.	A plate notation of an LDA model [BNJ03].	24
4.7.	Types of text summarization approaches. Graphic taken from: [Cha18]. . . .	26
4.8.	An example text and a weighted graph, obtained from it after TextRank was applied. The original image was taken from [MT04].	30
4.9.	An example text for keywords extraction. The original image was taken from [Ros+10].	31
4.10.	Keywords candidates. The original image was taken from [Ros+10].	32
4.11.	An example co-occurrence matrix, which was built using the cleaned terms. The original image was taken from [Ros+10].	32
4.12.	Scores for candidate expressions from Figure. The original image was taken from [Ros+10].	33
5.1.	A general project structure.	36
5.2.	The home page of the application.	37

5.3.	Graphical interface of the <i>topics</i> module.	39
5.4.	A piece of the discussion (<i>discussionID</i> = 4) used to generate topics.	40
5.5.	The parameter bar and four generated topics for the English discussion (<i>discussionID</i> = 4).	41
5.6.	Example of an error case when a user provides an invalid value for the <i>numTopics</i> parameter.	42
5.7.	Topics List API with the following parameters: <i>user</i> = test_user, <i>discussionID</i> = 4, <i>trainedModel</i> = false, <i>package</i> = sklearn, <i>inputModel</i> = NMF, <i>nGrams</i> = false, <i>numTopics</i> = 3	46
5.8.	Topics Retrieve API requesting a resource with the <i>primary key</i> = 24.	46
5.9.	Graphical interface of the <i>summary</i> module.	47
5.10.	The position (<i>positionID</i> = 320) from the <i>D-BAS</i> discussion in English which was later used to extract summaries for the evaluation step.	48
5.11.	Summary generated with the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = gensim, <i>ratio</i> = 0.7, <i>considerStructure</i> = false.	48
5.12.	Summary generated with the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = gensim, <i>ratio</i> = 0.3, <i>considerStructure</i> = false.	49
5.13.	Summary generated with the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = TF, <i>ratio</i> = 0.7, <i>considerStructure</i> = false.	49
5.14.	Example of an error case when a user provides an invalid value for the <i>positionID</i> parameter.	50
5.15.	An example of a position structure.	51
5.16.	Summary List API with the following parameters: <i>user</i> = test_user, <i>positionID</i> = 320, <i>considerStructure</i> = false, <i>inputModel</i> = gensim, <i>ratio</i> = 0.5.	53
5.17.	Summary Retrieve API requesting a resource with the <i>primary key</i> = 160.	53
5.18.	Graphical interface for the input parameters of the <i>keywords</i> module.	54
5.19.	Keywords extracted using the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = Rake, <i>ratio</i> = 0.3, <i>considerStructure</i> = false.	55
5.20.	Keywords extracted using the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = Rake, <i>ratio</i> = 0.6, <i>considerStructure</i> = false.	55
5.21.	Keywords extracted using the following parameters: <i>positionID</i> = 320, <i>inputModel</i> = TF, <i>ratio</i> = 0.6, <i>considerStructure</i> = false.	56
5.22.	Example of the error handling when an invalid <i>inputModel</i> parameter is provided.	56

5.23. Keywords List API with the following parameters: <i>user = test_user, positionID = 320, considerStructure = false, inputModel = rake, ratio = 0.5</i>	58
5.24. Keywords Retrieve API requesting a resource with the <i>primary key = 160</i>	58
6.1. An example exercise to evaluate an LDA model of the <i>gensim</i> package with three topics.	63
6.2. An example task to evaluate the pre-trained <i>gensim</i> LDA model with 50 topics (English, <i>gensim</i>).	64
6.3. An example task to evaluate the pre-trained <i>gensim</i> LDA model with 50 topics (German, <i>gensim</i>).	64
6.4. LDA — five topics with a pre-trained model (English, <i>gensim</i>).	65
6.5. LSI — five topics with a pre-trained model (English, <i>gensim</i>).	65
6.6. NMF — five topics with a pre-trained model (English, <i>sklearn</i>).	65
6.7. LDA — five topics with a pre-trained model (English, <i>sklearn</i>).	65
6.8. LSI — five topics with a pre-trained model (English, <i>sklearn</i>). On these scales, 1 stands for “Doesn’t fit at all“ and 10 stands for "Fits perfectly". . . .	65
6.9. LDA — five topics with a pre-trained model (German, <i>gensim</i>).	66
6.10. LSI — five topics with a pre-trained model (German, <i>gensim</i>).	66
6.11. NMF — five topics with a pre-trained model (German, <i>sklearn</i>).	66
6.12. LDA — five topics with a pre-trained model (German, <i>sklearn</i>).	66
6.13. LSI — five topics with a pre-trained model (German, <i>sklearn</i>).	66
6.14. LDA — five topics without a pre-trained model (English, <i>gensim</i>).	67
6.15. LSI — five topics without a pre-trained model (English, <i>gensim</i>).	67
6.16. NMF — five topics without a pre-trained model (English, <i>sklearn</i>).	67
6.17. LDA — five topics without a pre-trained model (English, <i>sklearn</i>).	68
6.18. LSI — five topics without a pre-trained model (English, <i>sklearn</i>).	68
6.19. LDA — five topics without a pre-trained model (German, <i>gensim</i>).	68
6.20. LSI — five topics without a pre-trained model (German, <i>gensim</i>).	69
6.21. NMF — five topics without a pre-trained model (German, <i>sklearn</i>).	69
6.22. LDA — five topics without a pre-trained model (German, <i>sklearn</i>).	69
6.23. LSI — five topics without a pre-trained model (German, <i>sklearn</i>).	70
6.24. The position (<i>positionID = 320</i>) from the <i>D-BAS</i> discussion in English which was later used to extract summaries for the evaluation step.	70
6.25. The position (<i>positionID = 50</i>) from the <i>D-BAS</i> discussion in German which was later used to extract summaries for the evaluation step.	71

6.26. An example task to rank the summary produced by the <i>gensim</i> package (English).	71
6.27. Evaluation results of the summary generated by the <i>gensim</i> package (English).	72
6.28. Evaluation results of the summary generated by the <i>gensim</i> package (German).	72
6.29. Evaluation results of the summary generated by the <i>TextRank</i> algorithm (English).	72
6.30. Evaluation results of the summary generated by the <i>TextRank</i> algorithm (German).	72
6.31. Evaluation results of the summary generated by the <i>TF</i> algorithm (English).	73
6.32. Evaluation results of the summary generated by the <i>TF</i> algorithm (German).	73
6.33. Evaluation results of the summary generated by the <i>TF-IDF</i> algorithm (English).	73
6.34. Evaluation results of the summary generated by the <i>TF-IDF</i> algorithm (German).	73
6.35. An example task to rank the keywords generated by the <i>Rake</i> algorithm (English).	78
6.36. Evaluation results of the keywords generated by the <i>gensim</i> package (English).	79
6.37. Evaluation results of the keywords generated by the <i>gensim</i> package (German).	79
6.38. Evaluation results of the keywords generated by the <i>Rake</i> algorithm (English).	80
6.39. Evaluation results of the keywords generated by the <i>Rake</i> algorithm (German).	80
6.40. Evaluation results of the keywords generated by the <i>TF</i> algorithm (English).	80
6.41. Evaluation results of the keywords generated by the <i>TF</i> algorithm (German).	80

List of Tables

5.1.	Set of training parameters (<i>sklearn</i>).	43
5.2.	Set of training parameters (<i>gensim</i>).	44
6.1.	Topic parameters for evaluation.	62
6.2.	Average ROUGE scores for generated English summaries.	74
6.3.	Average ROUGE scores for generated English summaries considering the position structure.	75
6.4.	Average ROUGE scores for generated German summaries.	76
6.5.	Average ROUGE scores for generated German summaries considering the position structure.	77
6.6.	Average ROUGE scores for generated English keywords.	81
6.7.	Average ROUGE scores for generated German keywords.	81
E.1.	Human-produced reference summaries gathered during the evaluation step.	120
F.1.	The keywords generated for the evaluation with the following parameters: positionID = 320 (English) and positionID = 50 (German), ratio = 0.5, considerStructure = false.	121
F.2.	The reference keywords given by the participants of the evaluation.	122

Bibliography

- [10k] *Ten Thousand German News Articles Dataset*. Accessed: 2020-08-15. URL: <https://tblock.github.io/10kGNAD/>.
- [All+17] Mehdi Allahyari et al. “Text Summarization Techniques: A Brief Survey”. In: *International Journal of Advanced Computer Science and Applications* 8.10 (2017). ISSN: 2158107X. DOI: 10.14569/ijacsa.2017.081052. arXiv: 1707.02268.
- [Bax58] P. B. Baxendale. “Man-made index for technical literature: an experiment”. In: *IBM Journal of Research and Development* 2(5) (1958), pp. 354–561.
- [BCD10] David Blei, Lawrence Carin, and David Dunson. “Probabilistic topic models”. In: *IEEE Signal Processing Magazine* 27.6 (2010), pp. 55–65. ISSN: 10535888. DOI: 10.1109/MSP.2010.938079.
- [BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. “LDA-blei.pdf”. In: 3 (2003), pp. 993–1022. ISSN: 0003-6951. DOI: 10.1162/jmlr.2003.3.4-5.993. arXiv: 1111.6189v1.
- [BP98] S. Brin and L. Page. “The anatomy of a large-scale hypertextual Web search engine BT - Computer Networks and ISDN Systems”. In: *Computer Networks and ISDN Systems* 30.1-7 (1998), pp. 107–117. ISSN: 01697552. DOI: 10.1016/S0169-7552(98)00110-X. arXiv: 1111.6189v1. URL: [http://dx.doi.org/10.1016/S0169-7552\(98\)00110-X](http://dx.doi.org/10.1016/S0169-7552(98)00110-X){\%}5Cnhttp://apps.webofknowledge.com/full{_}record.do?product=UA{\&}search{_}mode=GeneralSearch{\&}qid=6{\&}SID=X1pOWPMuSmOv1SlwJ6f{\&}page=1{\&}doc=2.
- [Cha18] Kushal Chauhan. *Unsupervised Text Summarization using Sentence Embeddings*. 2018.
- [Dew] *German Wikipedia dumps*. Accessed: 2020-08-15. 2020. URL: <https://dumps.wikimedia.org/dewiki/>.

- [DLB20] Anna De Liddo and Michelle Bachler. *DebateHub*. 2020. URL: <https://debatehub.net/>.
- [Enw] *English Wikipedia dumps*. Accessed: 2020-08-15. URL: <https://dumps.wikimedia.org/enwiki/20200801/>.
- [Gav] *Gavagai*. Accessed: 2020-08-15. URL: <https://www.gavagai.io/text-analytics/topic-modelling/?gclid=EAIaIQobChMIiuapnMbX6wIVGBwE>.
- [GK14] Vairaprakash Gurusamy and Subbu Kannan. “Preprocessing Techniques for Text Mining”. In: Oct. 2014.
- [Gpt] *OpenAI API*. Accessed: 2020-09-03. URL: <https://openai.com/blog/openai-api/>.
- [Gro] Hazy Research Group. *Probabilistic inference and factor graphs*. Accessed: 2020-08-15. URL: <https://deepdive.stanford.edu/inference#:~:text=Probabilistic%20inference%20is%20the%20task,event%20that%20John%20has%20cancer..>
- [HMK12] Yashodhara Haribhakta, Arti Malgaonkar, and Parag Kulkarni. “Unsupervised topic detection model and its application in text categorization”. In: *ACM International Conference Proceeding Series* (2012), pp. 314–319. DOI: 10.1145/2381716.2381775.
- [Hof99] Thomas Hofmann. “Probabilistic latent semantic indexing”. In: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1999* 51.2 (1999), pp. 50–57. DOI: 10.1145/312624.312649.
- [KI08] Mark Klein and Luca Iandoli. “Supporting Collaborative Deliberation Using a Large-Scale Argumentation System: The MIT Collaboratorium”. In: *SSRN Electronic Journal* (Feb. 2008). DOI: 10.2139/ssrn.1099082.
- [Kia] *Kialo*. Accessed: 2020-08-15. URL: <https://www.kialo.com/>.
- [Kle10] Mark Klein. “Using Metrics to Enable Large-Scale Deliberation”. In: *IN Convertino G Grasso A Millen DR De Michelis G Chi E H 2010 Group 2010 Workshop Luppicini* (2010). DOI: 10.1177/004057368303900411. URL: <http://scholar.google.com/scholar?hl=en{\&}btnG=Search{\&}q=intitle:Using+Metrics+to+Enable+Large+Scale+Deliberation{\#}0>.

-
- [KMM17] Tobias Krauthoff, Christian Meter, and Martin Mauve. “Dialog-based online argumentation: Findings from a field experiment”. In: *CEUR Workshop Proceedings 2012* (2017), pp. 85–99. ISSN: 16130073.
- [KR10] Werner Kunz and Horst W.J. Rittel. “Issues as elements of information systems”. In: *The Universe of Design: Horst Rittel’s Theories of Design and Planning* 9780203851586.131 (2010), pp. 181–185. DOI: 10.4324/9780203851586.
- [Kra+18] Tobias Krauthoff et al. “D-BAS - A dialog-based online argumentation system”. In: *Frontiers in Artificial Intelligence and Applications* 305 (2018), pp. 325–336. ISSN: 09226389. DOI: 10.3233/978-1-61499-906-5-325.
- [Kri+12] Travis Kriplean et al. “Supporting reflective public thought with ConsiderIt”. In: Feb. 2012, pp. 265–274. DOI: 10.1145/2145204.2145249.
- [Liu+16] Lin Liu et al. “An overview of topic modeling and its current applications in bioinformatics”. In: *SpringerPlus* 5.1 (2016). ISSN: 21931801. DOI: 10.1186/s40064-016-3252-8.
- [LS01] Daniel Lee and Hyunjune Seung. “Algorithms for Non-negative Matrix Factorization”. In: *Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*. 2001, pp. 556–562.
- [LZ10] Zhenxiao Li and Liqing Zhang. “Affine Invariant Topic Model for Generic Object Recognition”. In: *Advances in Neural Networks - ISNN 2010. Lecture Notes in Computer Science*. Vol. 6064. 2010, pp. 152–161. ISBN: 978-3-642-13317-6. DOI: https://doi.org/10.1007/978-3-642-13318-3_20.
- [MT04] Rada Mihalcea and Paul Tarau. “TextRank: Bringing Order into Text”. In: *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*. Barcelona, Spain: Association for Computational Linguistics, July 2004, pp. 404–411. URL: <https://www.aclweb.org/anthology/W04-3252>.
- [Nat] *NaturalText*. Accessed: 2020-08-15. URL: <https://www.naturaltext.com/>.
- [Net20] Group Computer Networks. *D-BAS*. 2020. URL: <https://dbas.cs.uni-duesseldorf.de/discuss/town-has-to-cut-spending#graph>.

- [NS15] Thien Hai Nguyen and Kiyooki Shirai. “Topic modeling based sentiment analysis on social media for stock market prediction”. In: *ACL-IJCNLP 2015 - 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, Proceedings of the Conference 1* (2015), pp. 1354–1364. DOI: 10.3115/v1/p15-1131.
- [RCL16] Toqir A. Rana, Yu N. Cheah, and Sukumar Letchmunan. “Topic modeling in sentiment analysis: A systematic review”. In: *Journal of ICT Research and Applications* 10.1 (2016), pp. 76–93. ISSN: 23375787. DOI: 10.5614/itbj.ict.res.appl.2016.10.1.6.
- [Rea20] ReasoningLab. *bCisive*. 2020. URL: <https://www.bcisiveonline.com/>.
- [Red] *Reddit*. Accessed: 2020-09-03. URL: <https://www.reddit.com/>.
- [Ros+10] Stuart Rose et al. “Automatic Keyword Extraction from Individual Documents”. In: Mar. 2010, pp. 1–20. ISBN: 9780470689646. DOI: 10.1002/9780470689646.ch1.
- [SB88] Gerard Salton and Christopher Buckley. “Term-weighting approaches in automatic text retrieval”. In: *Information Processing & Management* 24.5 (1988), pp. 513–523. DOI: 10.1016/0306-4573(88)90021-0. URL: <https://doi.org/10.1016%2F0306-4573%2888%2990021-0>.
- [Sch+17] Nadine Schneider et al. “Chemical Topic Modeling: Exploring Molecular Data Sets Using a Common Text-Mining Approach”. In: *Journal of Chemical Information and Modeling* 57.8 (2017), pp. 1816–1831. ISSN: 15205142. DOI: 10.1021/acs.jcim.7b00249.
- [Sem] *Semantria*. Accessed: 2020-08-15. URL: <https://sourceforge.net/software/product/Semantria/>.
- [Suh+16] Sangho Suh et al. *L-EnsNOMF: Boosted Local Topic Discovery via Ensemble of Nonnegative Matrix Factorization*. 2016. URL: https://sanghosuh.github.io/lens_nmf-icdm/#/3/2.
- [SVB07] David Schneider, Christian Voigt, and Gregor Betz. “Argunet – A software tool for collaborative argumentation analysis and research”. In: *CMNA VII - Comput Models Nat Arg* 10 (Jan. 2007).

-
- [SVL14] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. “Sequence to Sequence Learning with Neural Networks”. In: *CoRR* abs/1409.3215 (2014). arXiv: 1409.3215. URL: <http://arxiv.org/abs/1409.3215>.
- [WG17] Douglas Walton and Thomas Gordon. “Argument Invention with the Carneades Argumentation System”. In: *SCRIPT-ed* 14 (Dec. 2017), pp. 168–207. DOI: 10.2966/scrip.140217.168.
- [WGG17] Haibing Wu, Xiaodong Gu, and Yiwei Gu. “Balancing between over-weighting and under-weighting in supervised term weighting”. In: *Information Processing and Management* 53.2 (2017), pp. 547–557. ISSN: 03064573. DOI: 10.1016/j.ipm.2016.10.003.
- [Wu+07] Chunguo Wu et al. “Machine learning-based keywords extraction for Scientific Literature”. In: *Journal of Universal Computer Science* 13.10 (2007), pp. 1471–1483. ISSN: 0958695X.
- [ZZC14] Weizhong Zhao, Wen Zou, and James J. Chen. “Topic modeling for cluster analysis of large biological and medical datasets”. In: *BMC Bioinformatics* 15.11 (2014), pp. 1–11. ISSN: 14712105. DOI: 10.1186/1471-2105-15-S11-S11.
- [Bel05] J. R. Bellegarda. “Latent semantic mapping [information retrieval]”. In: *IEEE Signal Processing Magazine* 22.5 (2005), pp. 70–80.
- [Luh58] H. P. Luhn. “The Automatic Creation of Literature Abstracts”. In: *IBM Journal of Research and Development* 2.2 (1958), pp. 159–165.

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Masterarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Düsseldorf, 28. September 2020

Olga Batiukova

