Comparison of approaches for detecting topical differences of text documents

Bachelor's Thesis

Chair of Data Science

Dr. Christin Seifert

Faculty of Computer Science and Mathematics

University of Passau

by

Stefanie Urchs

Erklärung

Hiermit erkläre ich, dass ich die Arbeit selbstständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel benutzt, sowie wörtliche und sinngemäße Zitate auch als solche gekennzeichnet habe.

Passau, den

Datum Unterschrift

Abstract

Information retrieval and text mining textbooks like "Introduction to Information Retrieval" [MRS08] and "Text Mining Handbook" [Fel06] contain several methods for feature representation, text summarisation and text categorisation. All methods either reduce texts to their most relevant concepts or seek out similarities between texts or within a single text. But non of these methods analyse the differences between texts. The extraction of differences between texts can be a useful tool for a quicker understanding of a whole topic, because texts concerning a similar topic will focus on different subtopics. This thesis is concerned with detecting topical differences between text documents and tries to answer the following question:
"How to extract information from texts to qualitatively summarise their content difference?"
To answer this question this thesis explores the related research. Feature representation and text summarisation provide promising approaches that could be used as a basis for this thesis. Specifically the term frequency and inverse document frequency (tf-idf) computation, the TextRank algorithm and the latent Dirichlet allocation (LDA) are chosen. These are part of a pipeline in which two texts are preprocessed, the most interesting concepts of the texts retrieved, concepts compared, similarities removed, and finally the results presented to the user. The implementation is evaluated quantitatively using automatic means, as well as qualitatively by humans. The thesis follows the underlying idea that if all similarities are removed, only differences remain. All approaches retrieve differences, but the description of the source text and the quality of the retrieved differences could be improved.

Contents

1. Introduction

When searching the web for a certain topic thousands of articles, books and papers are provided. It is no longer humanly feasible to manually review the whole amount of information any more. Because of this, computers are needed to process texts. With text representation techniques, texts can be processed by computers. Furthermore, text summarisation techniques can help to get an overview of different texts. These two techniques are largely utilised to find similarities between multiple texts or between a text and a query [MRS08]. The extraction of differences between texts can be a useful tool for a quicker understanding of a whole topic. Texts concerning a similar topic will focus on different subtopics, which can be can be extracted by finding the differences between text documents. Currently most information retrieval approaches can not be applied to find the differences between text documents. This thesis develops several approaches to differentiate texts and evaluates them.

1.1. Objective

The research question of this thesis is:

> "How to extract information from texts to qualitatively summarise their content difference?"

This thesis is concerned with differentiating two texts. The qualitative content difference deals with which concepts are part of the first text but not of the second one. The qualitative aspect emphasises that semantic differences and not syntactic ones should be retrieved.
The approaches that are developed in this thesis will extend the following approaches:

- vector space model
- topic model
- text graph

1.2. Structure

The following chapters are structured as described below:

- **Related Work**
  This chapter discusses the related work that this thesis is based on.
- **Approach**
  The approach chapter discusses the methodology to analyse topical differences of text documents independently of a programming language.
- **Implementation**
  In this chapter the code of the approaches to compute the differences is described.
- **Evaluation**
  The evaluation consists of an automated and a human evaluation.
  In the automated evaluation the approaches of the implementation chapter are tested against certain benchmarks.
  In the human evaluation humans are asked to evaluate the quality of the approaches discussed in the implementation chapter.
- **Conclusion**
  The conclusion summarises the thesis and discusses some implications and limitations of the approaches explored. Finally some possible areas for future research are suggested.

2. Related Work

This chapter describes the previous research done in this area. These include a general introduction to preprocessing followed by specific approaches to feature representation, text summarisation, and text categorisation. Moreover the following algorithms are discussed in detail: tf-idf, LDA and TextRank.

2.1. Preprocessing

Before texts can be used for computation, usually some preprocessing tasks are preformed. These tasks are traditionally stopword removal, stemming or lemmatization, handling of punctuation, special characters and digits and the case of letters [Liu11].
Stopwords are words that occur frequently in a language but are insignificant. Some stopwords in the English language are: a, an, by, in, of, that, the, to, when, who, with. These words should be removed because they only help constructing the document but do not represent any content of the text [Liu11].
The goal of both stemming and lemmatization is to reduce words to a common base form, but they differ in the execution. Stemming is a heuristic process that cuts off the ends of words. In contrast to this, lemmatization uses a vocabulary and a morphological analysis of words to return the word base or dictionary form of the word [MRS08].
The handling of punctuation, special characters and digits depends on the domain the preprocessing is used in. In traditional information retrieval systems, numbers are removed in the preprocessing. This does not include dates, times and other specified types. For special characters and punctuation, specific rules are used. The case of the letters is usually unified to either all lower or all upper case [Liu11].

2.2. Feature Representation

The choice of features is a big factor for the performance of machine learning methods. As long as modern machine learning algorithms are not able to extract and organise the most important informations by themselves, feature representation is labour-intensive [BCV13].

2.2.1. Vector Space Model

The main concept of the vector space model (VSM) is to represent each document as a vector in a vector space (consequently as a matrix).
Vectors with a big difference are semantically more different than vectors with a small difference. VSM has a underlying distributional hypothesis. This hypothesis states that words with similar meanings will occur in similar contexts. Because of this hypothesis, it is possible to measure the similarity of meaning in documents [TP10].
The standard way to quantify the similarity between document vectors is to compute their cosine similarity. The cosine similarity divides the dot product of two vectors by their Euclidian length. This approach compensates for the length of the documents [MRS08].

term frequency and inverse document frequency (tf-idf)
The tf-idf computation is the most common approach to generate vectors.
This approach first computes how often a term occurs in a document (term frequency). Then the inverse of how often the term occurred in all viewed documents is computed (inverse document frequency). Finally these two measurements are combined into a vector [MRS08].

2.2.2. Language Model

The language model is a statistical model that tries to estimate a model for the word distribution within text documents. In other words language models are functions that draw strings from texts and assign probabilities to them. The simplest form of a language model is the unigram language model in which each term is considered individually. More complex forms exists, such as the bigram language model which lays conditions on the previous retrieved term [MRS08].

2.3. Text Summarisation

The rapid growth of the internet resulted in a massive increase in available information. To cope with this efficiently, human language technologies such as text summarisation were invented. Their aim is to condense the source text, by selecting what is important in the source, to a summary of the source text. Thus the user can save time and resources when looking for specific information within documents [LP12].

2.3.1. Topic Models

Topic models assume that texts are constructed with a mixture of topics in mind. These topics are probability distributions over all words in the corpus.
First a set of topics is modelled over the entire corpus. In a second phase the topic probabilities are adjusted based on the individual texts. The topics are often represented by the word that has the highest probability in a given topic. Texts are represented as a probability distribution over all topics. Only the topics with the highest probability distribution are assigned to a document [Ale14].
In the end, topic modelling is a dimensionality reduction by projecting each document to a low-dimensional feature space (the topics).

latent Dirichlet allocation (LDA)
On an abstract level, LDA consists of two steps. In the first step all unique words, in all considered texts, are assigned to a random topic. In the second step the distribution of words in the topics is improved by going through every text. Here the probability for every word is updated [BL09].

2.3.2. Text Graphs

To generate a graph from a text, the following four steps are necessary: first the text units that are best suited for the task are identified and added as nodes to the graph. Secondly the relations between these nodes are identified and used to draw edges between the nodes. These edges can be directed or undirected and weighted or unweighted. The third step is to itteratly apply the graph-based ranking algorithm until it converges. Finally the nodes are sorted based on their final score. Additionally, the values attached to the nodes are used for ranking or selection decisions [MT04].
Text graphs are already used for difference extraction. Marta Gruszecka and Michal Pikusa published a paper about a comparative study of newspaper articles about the 2010 Polish Air Force presidential plane crash [GP15].

TextRank
TextRank represents individual words as nodes. The edges between these nodes consist of the co-occurrence of these words within a window of n words. The nodes are ranked by how often they occur in co-occurrence. The output of TextRank are the highest-ranked nodes [MT04].

2.3.3. Extractive Summaries

Extractive summarisation methods identify the most important parts of a text. From these parts sentences are extracted to compose a summary of the underlying text. All summarisation methods perform the following three tasks:

1. Construction of an representation of the input text, which expresses the main aspects of the text. This representation consists of sentences.
2. Based on this representation carry out a ranking of the sentences.
3. Select the highest ranked sentences and combine them to a summary.

[APA+17]

2.3.4. Machine Learning Methods

Machine learning methods employ statistical techniques to summarise texts. Some methods rely on naive-Bays and assume that the used features are independent. Other methods concentrate more on machine learning algorithm that make no independence assumption, these methods also concentrate on the choice of appropriate features. Furthermore there are methods that incorporate hidden Markov models and log-linear models to improve the summarisation. A recent development is to use neuronal networks and third party features, like words that commonly occur in search engine queries, to improve extractive text summaries [DM07].

2.4. Text Categorisation

Due to the increased availability of digital documents, content-based document management tasks, also known as information retrieval, have gained a prominent status in the information systems field. Text categorisation is one of these tasks.
In text categorisation natural language is labelled with thematic categories. Text categorisation is generally applied in any application requiring document organisation or selective and adaptive document dispatching [Seb02].

2.4.1. Classification

Classification (or categorisation) of elements is one of the most common themes in analysing complex data. The main objective is to assign a element to a predefined set of categories [Fel06]. Classification consists of single label and multi label classification. The goal of single label classification is to assign a element to one predefined category, or to associate it with one label. For automatic text categorisation multi label classification is used. In multi label classification a element can be associated to more than one label [Sor10].

2.4.2. Clustering

In contrast to classification clustering sorts elements into groups without prior definition of these groups. Every decision concerning the grouping (clustering) of the elements is made by analysing the given data [Fel06].

2.5. Discussion

To accurately capture the frequency and context of words in a text, different forms of a word need to be reduced to a common base. Two common methods for this are stemming and lemmatization. Stemming can incur two types of errors: in the case of over-stemming two words with different stems are stemmed to the same root. Conversely under-stemming is the case where words are reduced to different roots, even if they should be stemmed to the same one [J^+11].
These errors can be reduced by using lemmatization, because it uses a vocabulary and a morphological analysis of words to return the word base or dictionary form of the word [MRS08].
All following approaches are selected depending on whether a Python implementation exists or not. For each approach an implementation is provided in the gensim library which is used in this thesis.
For the VSM, the tf-idf computation is used. This should act as the baseline for this thesis. The area of text summarisation is represented by the LDA and TextRank. No approaches from the area of text categorisation are used. This thesis works with unseen texts, so there are no predefined categories in which a element could be categorised.
Thus the usage of classification is impossible. Clustering needs certain input features. From the approaches discussed above, only the vectors of the VSM would lead to meaningful results, but clustering these would restrain the incoming features to vectors of the whole documents. In other words there would only be two features to cluster. A clustering on two features would not lead to a meaningful result.

3. Approach

Based on the related work, the tf-idf computation, the TextRank algorithm and LDA are chosen as underlying approaches for this thesis. Figure 1 shows the pipeline all approaches follow.
First, all texts are preprocessed. Afterwards the text differences are obtained. It would be possible to create an approach that directly displays the differences from the source texts. But in this thesis this differentiation is split in two parts. At first the most meaningful concepts are extracted. Afterwards all similarities between the concepts are removed. Thus only the differences remain. Finally the result of the difference computation is displayed.

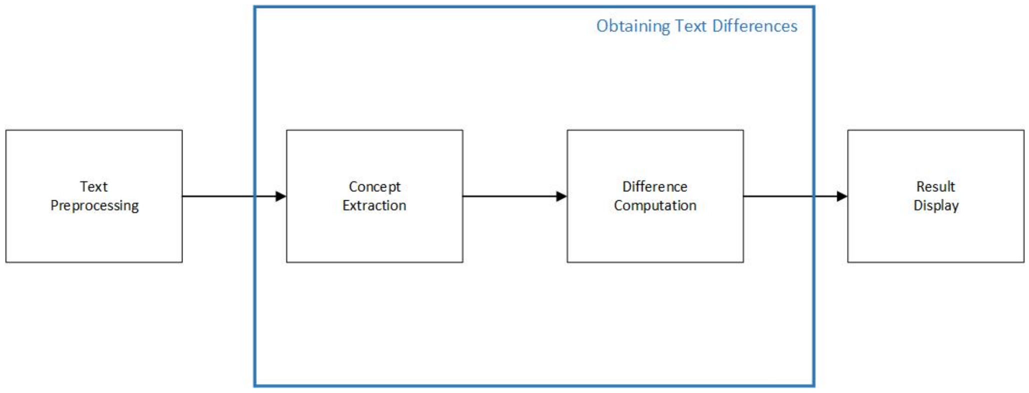


Figure 1: Preprocessing pipeline that is followed by all approaches

Two same-length texts are randomly chosen from the Reuters-21578 corpus to exemplify the results provided:

Text One:

> *President Reagan's approval rating fell after the Tower Commission criticised his handling of the Iran arms scandal, a New York Times/CBS poll indicates.*
> *The poll found 51 pct of those surveyed thought he was lying when he said he did not remember if he had approved the original arms sales to Iran and 35 pct thought he was telling the truth.*
> *The poll found 42 pct of those surveyed approved Reagan's handling of his job and 46 pct disapproved. The approval rating was the lowest since January 1983, when 41 pct approved of the way Reagan was doing his job.*

Text Two:

> *Egypt allowed five fugitive Libyan soldiers who landed in a military plane in the far south of the country last night to stay and flew them to Cairo, official sources said.*
> *It appeared the government had agreed to demands by the five – two officers and three privates – for political asylum but there was no immediate announcement.*
> *Official sources said Egyptian servicemen flew the Libyans north from Abu Simbel in their C-130 transport plane. The status of the sixth Libyan on board, the pilot, was not immediately known.*

3.1. Preprocessing

For an optimal result certain preprocessing steps are necessary.

- tokenisation
- conversion to lower case
- stopword removal
- lemmatisation

The tokenisation splits the text into single words. The next step converts all words to lower case. This step is necessary to map words from the beginning of a sentence to their counterparts in the rest of it. Hereafter all words that are common in the English language and carry minor meaning by themselves, the stopwords, are removed. These words do not contribute to the overall meaning of the text. In the end all words are lemmatised. This means that all words are reduced to their basic word form [BKL14].
Later on, all approaches work with a bag-of-words approach. To turn a text into a bag-of-words, every occurrence of a unique word is counted. In the end only the unique words and their count are saved. By lemmatising words, all words with the same basic word form are recognised as one unique word. To do all of this the tokenisation from a above is necessary [BKL14].
To to reduce words to their base form it would also be possible to stem the words. One of the most popular stemming algorithms is the "porter stemmer". This stemmer removes suffixes from words considering certain rules. Since only suffixes are removed certain words can not be recognised, e. g. deceive/deception, resume/resumption or index/indices [Por80].
Lemmatization is preferred to stemming because it tries to understand the part of speech (POS) and the context in the given sentence of the source word to reduce it to its root form [J$^+$11].

This behaviour helps to reduce under-stemming and over-stemming errors.
In A.1.1 on page 52 the preprocessing steps are applied to text one.

3.2. Concept Extraction

After the preprocessing, the most important text concepts are extracted from the tokens. To extract these concepts, each approach works differently. The preprocessing above ensures that all approaches concentrate on the most meaningful words.

tf-idf
To start the tf-idf computation, two texts have to be provided, along with a ratio of how many words in regard to unique words in each text should be retrieved and a comparative method. Tf-idf computes the tf-idf values for both texts. Afterwards the words with the highest tf-idf value are extracted. The number of words is computed out of the ratio given by the user and the number of unique tokens in each text.
A possible default parameter for the ratio of best words is one third of the unique words. A similar parameter is used in the TextRank paper that decides the number of keywords depending on the number of nodes in the text graph, which results out of unique tokens in the text [MT04].
A classic tf-idf computation would not involve elaborate preprocessing as described above. But it was deemed beneficial to use stopword removal and lemmatization to concentrate the computation on the most important words.
The comparative method is used for the computation of differences between the source texts.
In A.1.2 on page 53 the tf-idf computation is applied to text one and text two. The ratio of best words is set to the default parameter of 0.33. Because of that the 33% of the unique words with the highest tf-idf value in each text are extracted.

TextRank
Like in the tf-idf, the user has to provide two texts, a ratio for the text and a comparative method as parameters for the algorithm. The text ratio describes what percentage of nodes of the text graph should be retrieved. The nodes with the highest number of edges are retrieved.
According to the TextRank paper a third of the nodes in the text graph is a fitting default parameter for the text ratio [MT04].
TextRank could be used without preprocessing. But the usage of preprocessing leads to better results in regards to the meaningfulness of retrieved words.
In A.1.3 on page 54 the TextRank algorithm is applied to text one and two with a text ra-

tio of 0.33. TextRank retrieves fewer words than the tf-idf from the texts because not all tokens are nodes. If a node has too few edges it is dropped from the text graph [MT04].

LDA
The LDA computation needs two texts and a number of topics as input parameters for the algorithm.
The number of topics will be distributed over both texts. The LDA then computes the best-fitting topics for each text.
The LDA paper suggests no default parameter for the number of topics [BNJ02].
Preprocessing is used in this approach to enhance the relevance of the topics.
A exemplary usage of the LDA on text one and two with five topics can be found in A.1.4 on page 54. The number five for the topics is computed as default parameter in the implementation chapter. Even if for the whole corpus five topics are computed the probabilities for topic one, two and four are too small to be assigned to a single text. Because of this text one and text two only get one topic assigned. If the topic parameter is reduced the computed topics get less differentiated. Because of this both texts get topics assigned that contain words from the other text. If the topics parameter is increased more topics with low probabilities are computed and never assigned.
After the most relevant concepts are extracted from the texts, all approaches have two lists with the concepts — one list for every source text. These lists are the input for the next step, the computation of the differences.

3.3. Difference Extraction and User Presentation

After the most important concepts are extracted from the texts, the similarities are removed. By erasing everything that is similar only the differences remain, which are then presented to the user.

tf-idf and TextRank
Both tf-idf and TextRank receive two lists of words with the most interesting text concepts. The lists can be compared in two ways. These two ways are the comparative methods the user has to choose when starting the computation. The first one is to simply compare the words one by one and remove all that are in both lists. This method is called simple differentiation. The second one compares whether a word is a synonym, hypernym or hyponym of a word from the other list. This one is called semantic differentiation. If words match the criteria from the comparative methods they are removed from the result list. The result list contains all interesting text concepts from the first text, minus the words the differentiation methods removed. This list is presented to the user.

In A.1.5 on page 55 the differences for tf-idf and TextRank with both comparative methods can be seen. As mentioned before the algorithms are applied to text one and two using a ratio of 0.33. The comparative methods do not produce notably different results. Furthermore the results for tf-idf and TextRank highly resemble one another. The length of result lists vary with the length of the source texts and the chosen ratio parameters.

LDA
The LDA returns a list of words for every topic. Every word of the first topic list is then compared to the words in the second one. If a word occurs in both lists it is removed. The user receives the result of this comparison.
The result of the LDA with five, two and nine topics on text one and text two can be found in A.1.5 on 55. As stated above, with less than five topics the result is empty because both topics were assigned to both texts. In addition both topics contain very similar words.
If nine topics are assigned over the corpus, the source texts get just one topic assigned because the probabilities for the other topics are too small.

User Interface
The user interacts with a graphical user interface (GUI) in which they could choose between these approaches and, if necessary, the two comparative methods. The result for all approaches is displayed under both texts that are entered for the computation.
The tf-idf and TextRank result are displayed as a list of words and the LDA results as a list of topics.

4. Implementation

The tf-idf, TextRank and LDA approaches and the preprocessing that are designed in the chapter 3 are implemented in this one. The implementation is written in Python 3.6 using the following libraries: `collections, gensim, heapq, json, matplotlib.pyplot, nltk, numpy, operator, os, pandas, random, re, timeit and tkinter.`
The `collection, gensim, heapq, nltk, operator, os` and `re` libraries are mostly used for the implementation of the approaches. The GUI of this thesis is implemented with the `tkinter` libary. The other libraries are mostly used for evaluation purposes.

4.1. Architecture

The thesis project consists of three directories:

- Approaches
  main functionality, computation of the approaches
- GUI
  User interface
- Test
  scripts for the evaluation of the approaches

The approaches directory contains the implementation of the approaches. Tf-idf, TextRank and LDA are each implemented in a different script, for clarity reasons. All methods that are used by one or more approaches are outsourced to the Util script.
The GUI directory contains the scripts for the user interface. These help the user to interact with the approaches in a more intuitive way.
Finally, in the Test directory, all scripts for the evaluation are located. These contain a testing GUI, automatic benchmark tests and the human evaluation.
Figure 2 shows the interaction between the Python scripts.

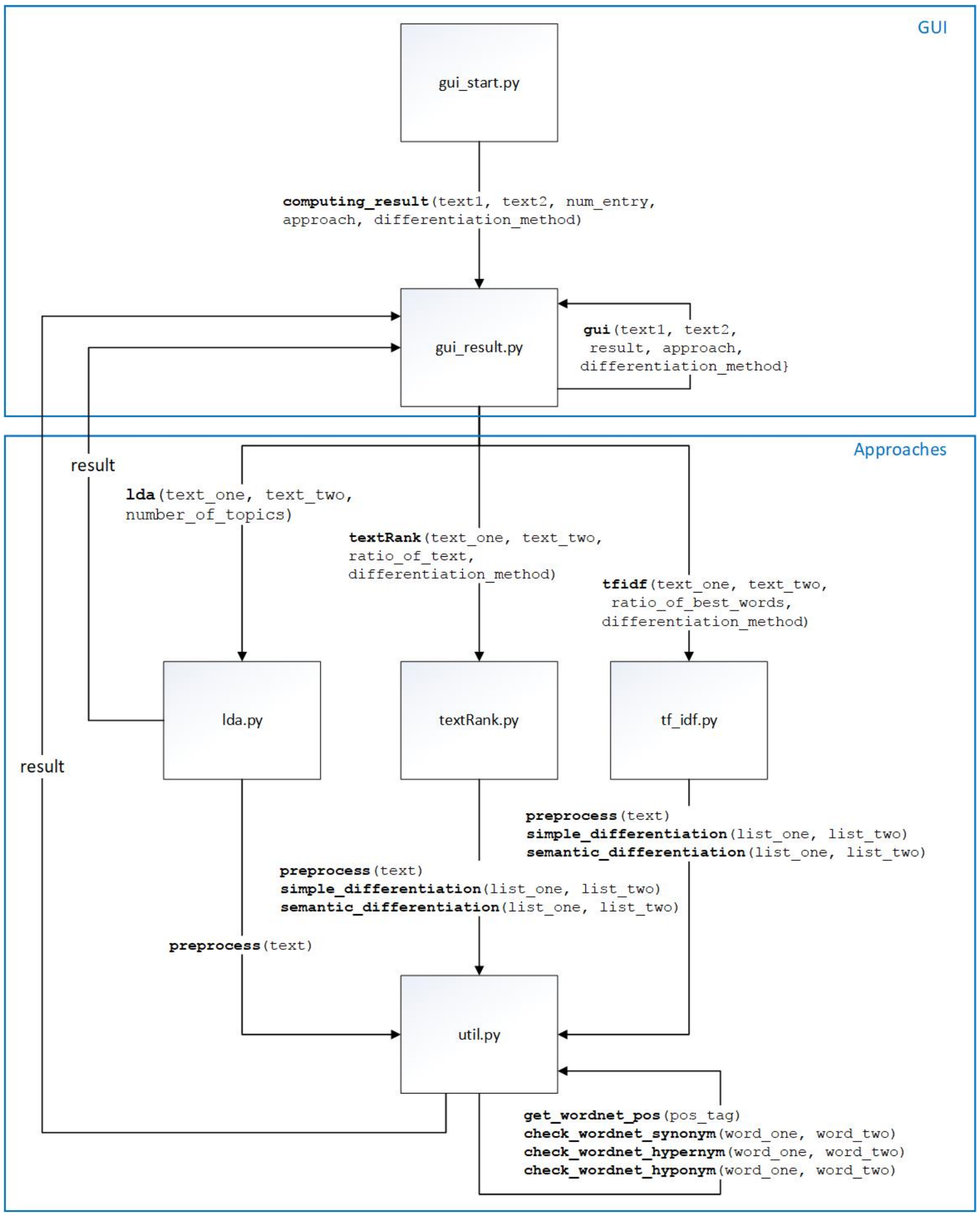

GUI
gui_start.py
computing_result(text1, text2, num_entry, approach, differentiation_method)
gui_result.py
gui(text1, text2, result, approach, differentiation_method}
Approaches
result
lda(text_one, text_two, number_of_topics)
textRank(text_one, text_two, ratio_of_text, differentiation_method)
tfidf(text_one, text_two, ratio_of_best_words, differentiation_method)
lda.py
textRank.py
tf_idf.py
result
preprocess(text)
simple_differentiation(list_one, list_two)
semantic_differentiation(list_one, list_two)
preprocess(text)
simple_differentiation(list_one, list_two)
semantic_differentiation(list_one, list_two)
preprocess(text)
util.py
get_wordnet_pos(pos_tag)
check_wordnet_synonym(word_one, word_two)
check_wordnet_hypernym(word_one, word_two)
check_wordnet_hyponym(word_one, word_two)

Figure 2: Relationship between the Python scripts of this thesis.

4.2. Code Description

In this chapter the code in the "Approaches" and "GUI" directories is discussed in detail. The content of the "Test" directory is discussed in the evaluation chapter.

4.2.1. Approaches

This directory contains the preprocessing, the extraction of most interesting concepts and the removal of similarities between the text files for each approach. For clarity purposes the approaches are split into separate scripts.

tf_idf.py

This script contains the method `tfidf(text_one, text_two, ratio_of_best_words, differentiation_method)`.

This method requires the following parameters:

- `text_one`: First text
- `text_two`: Second text
- `ratio_of_best_words`: Float that defines the n percent words, per document, with the highest tf-idf value that should be considered for the differences between the documents, n depends on the unique tokens in the text
- `differentiation_method`: number of the used differentiation method, two inputs are possible:
  - `simple`: This differentiation method corresponds to the simple differentiation that compares the n best words of the documents and returns all words that are not equal with a word of the other document
  - `semantic`: This differentiation method corresponds to the semantic differentiation that compares the n best words of both texts and removes all words that are in a synonym, hypernym or hyponym relationship. The remaining words are returned.

First the input variables are checked for invalid values. If an invalid value is detected, an error message is returned. Subsequently the input texts are preprocessed with the preprocess method from util.py. Next for every text the user-supplied `ratio_of_best_words` is used compute the actual number of tokens. Afterwards both preprocessed texts are merged into one corpus which is then converted to a bag-of-words. In this bag-of-words, a word ID is matched to the number of occurrences of this word. A dictionary with the word to ID mapping is saved. Then the tf-idf model is trained and applied to the whole corpus. Now every word ID has a tf-idf weight. The corpus is split back into two texts and the IDs are mapped back to words.

From every text the `ratio_of_best_words` words with the highest tf-idf value are extracted and saved to lists, without the tf-idf value (details can be found on page 56 in listing 1). Furthermore the lists are processed with either the simple or semantic differentiation. These differentiation methods are implemented in utils.py. Both differentiation methods return a list of words with the highest tf-idf rating for the first text, after cleaning the list (details can be found on page 56 in listing 2).

textRank.py
This script contains the method `textRank(text_one, text_two, ratio_of_text, differentiation_method)`.
TextRank requires the following parameters:

- `text_one`: First text
- `text_two`: Second text
- `ratio_of_text`: float that defines the number of extracted keywords as a percentage of all words in this text
- `differentiation_method`: number of the used differentiation method, two inputs are possible:
  - `simple`: This differentiation method corresponds to the simple differentiation that compares the n best keywords of the documents and returns all words that are not equal with a word of the other document
  - `semantic`: This differentiation method corresponds to the semantic differentiation that compares the n best keywords of both texts and removes all keywords that are in a synonym, hypernym or hyponym relationship. The remaining keywords are returned.

First all input variables are checked for invalid values. If an invalid value is detected, an error message is returned. Afterwards the input text is preprocessed with the preprocess method form utils.py and joined into a string again (details can be found on page 57 in listing 3).
Thereafter the keywords are extracted from the preprocessed text (details can be found on page 57 in listing 4).
Finally the keyword lists are preprocessed with either the simple or semantic differentiation. These differentiation methods are implemented in utils.py. Both differentiation methods return a list of keywords for the first text (details can be found on page 57 in listing 5).

lda.py
This script contains the method `lda(text_one, text_two, number_of_topics)`.
This method requires the following parameters:

- `text_one`: First text
- `text_two`: Second text
- `number_of_topics`: number of topics that should be generated from the corpus

First the input variables are checked for invalid values. If an invalid value is detected, an error message is returned. Subsequently the texts from text_one and text_two are preprocessed with the preprocess method from util.py.
Afterwards the texts are merged into one corpus and converted to a bag-of-words. On this bag-of-words the LDA model is trained. Thereafter both texts are turned into a bag-of-words on their own and the topics for each text are computed (details can be found on page 57 in listing 6). Usually the LDA is trained on a corpus of several hundred documents. This thesis could have used the whole Reuters-21578 corpus for the training. This is not done because the approach should work on every unseen corpus of texts, no prior known corpus exists. This could be archived by gradually adding the texts that the LDA is preformed on to a corpus. Doing this introduces the problem of continuously saving a increasing corpus.
Then the topic lists are converted back to words. For better readability the lists are cleaned: everything that is not a word is removed. This is necessary because when the topics are printed, they are separated by a plus sign and the probability of each word is listed. The next step is to preprocess the resulting words again because the cleaning leads to some artefacts that would otherwise be displayed as topics. Thereafter all words that are present in the topics of both the first and the second text are removed. Finally the topic list of the first text is returned (details can be found on page 58 in listing 7).

util.py
This script contains methods that are used by one or more approaches.

`preprocess(path)`
Tokenises the incoming text. Everything that is not a letter is removed and every word is converted to lower case. Next all tokens are tagged with a POS tag. This is necessary for lemmatization with wordnet at a later point in time. `nltk` only provides the treebank POS tags but these cannot be compiled by wordnet. To make wordnet read the POS tags, a conversion to the wordnet POS tags is required. Before the tokens are lemmatised all stopwords and all words without a POS tag are removed. Words shorter than three letters are also removed, as these are most likely tokenising artefacts. In the end, all remaining words are lemmatised with wordnet. This means these words are reduced to their basic

word form. These lemmas are returned as a list of words (details can be found on page 59 in listing 8).

`get_wordnet_pos(pos_tag)`
This method helps to convert the treebank POS tags to wordnet POS tags.
In wordnet four POS tags are defined, treebank knows thirty six. This method looks up the incoming treebank POS tag and maps it to the corresponding wordnet POS tag. If none is available the treebank POS tag is mapped to '' (details can be found on page 59 in listing 9).

`check_wordnet_synonym(word_one, word_two)`
This method checks if a word is a synonym of another. At first this method gets the wordnet synset for the first word. The synset contains all synonyms for the word. Afterwards the synset codes are converted to actual words. The next step is to check if the second word is part of the synonyms of the first one. In the end, a boolean, indicating whether the words are synonym or not, is returned (details can be found on page 60 in listing 10).

`check_wordnet_hypernym(word_one, word_two)`
This method checks if one word is a hypernym of another. This method opens the synset of the first word and collects all hypernyms to this synset code. After that the hypernym codes are converted to words and the second word is compared with them. If the second word matches a hypernym, True is returned, otherwise False (details can be found on page 60 in listing 11).

`check_wordnet_hyponym(word_one, word_two)`
This method works the same way as the hypernym method above. At first the synset code for the first word is opened and with that code all hyponyms of this word are collected. These hyponym synset codes are converted to words. Next the second word is compared with all the hyponyms of word one. If there is a match True is returned and False otherwise (details can be found on page 61 in listing 12).

`simple_differentiation(list_one, list_two)`
This method implements the simple differentiation. It is used by the tf-idf and TextRank approaches. It compares two lists and removes every word that is in both lists. A list of all remaining words of the first list is returned (details can be found on page 61 in listing 13).

`semantic_differentiation(list_one, list_two)`
This method implements the semantic differentiation. It is also used in the tf-idf and the TextRank methods. It removes all words that are a synonym, a hypernym or a hyponym of a word in the other list. Like in `simple_differentiation` a list of all remaining words of the first list is returned (details can be found on page 61 in listing 14).

4.2.2. GUI

This package contains the two GUI scripts. The GUI provides an intuitive way to apply the approaches to the input data and to get a clear display of the results.

gui_start
The script starts the dialogue in figure 3 in which the user can pick two files as well as one of the three approaches to compute the differences between the chosen texts.

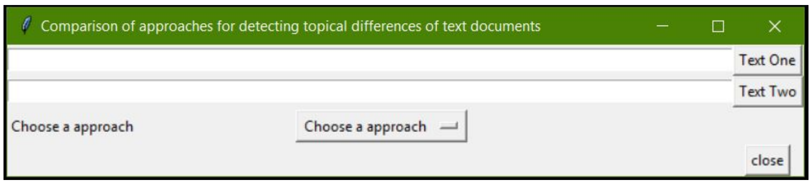
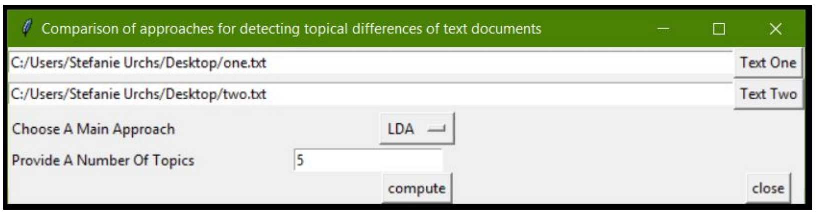

Comparison of approaches for detecting topical differences of text documents
Text One
Text Two
Choose a approach
Choose a approach
close

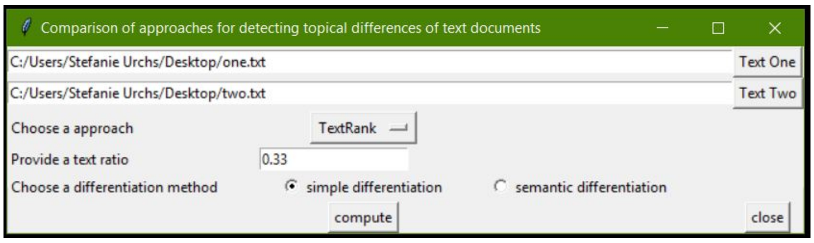
Figure 3: Default start GUI without input.

Depending on the chosen approach input fields for it are displayed. In figure 4 LDA is selected. Figure 5 shows the dialogue if TextRank is selected. If tf-idf is selected the window looks similar to the display of the TextRank.


Comparison of approaches for detecting topical differences of text documents

C:/Users/Stefanie Urchs/Desktop/one.txt — Text One
C:/Users/Stefanie Urchs/Desktop/two.txt — Text Two
Choose A Main Approach: LDA
Provide A Number Of Topics: 5

Figure 4: GUI after LDA was selected with two selected texts and a default value for the number of topics.


Comparison of approaches for detecting topical differences of text documents
Text One
Text Two
compute
close

C:/Users/Stefanie Urchs/Desktop/one.txt

C:/Users/Stefanie Urchs/Desktop/two.txt

Choose a approach: TextRank

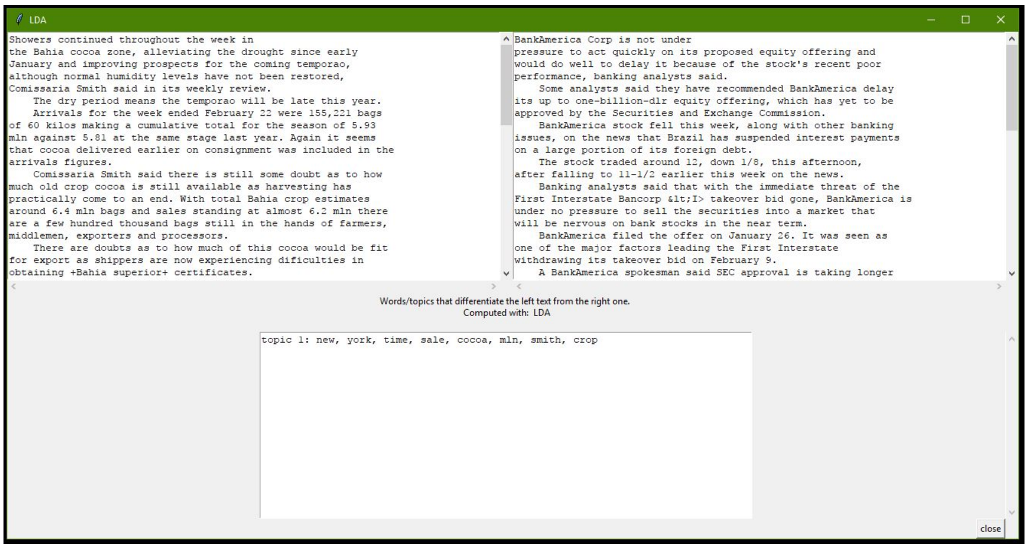
Provide a text ratio: 0.33

Choose a differentiation method: [✓ simple differentiation] semantic differentiation

Figure 5: GUI after TextRank selection with default parameters and two selected texts.

Clicking the compute button collects and processes all inputs and displays the result in a separate window.

gui_result
The result GUI displays the first text in the upper left and the second one in the upper right. Below the texts the result of the computation is shown.
Figure 6 shows that the result for LDA displays all words of a topic in one line. Every new line is a new topic.
In figure 7 the TextRank result is displayed. Every keyword is displayed an individual line. The tf-idf result is displayed in the same way.



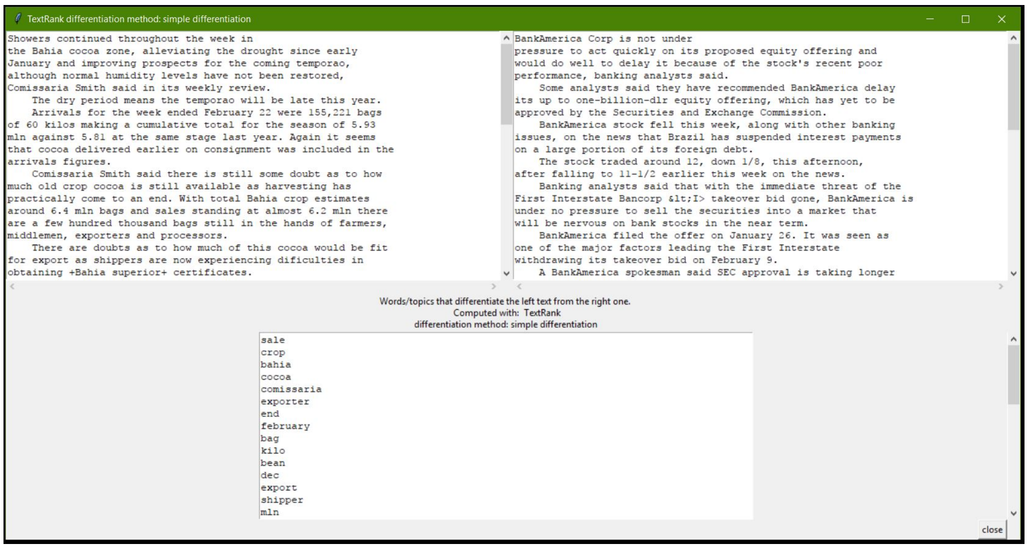
```
Showers continued throughout the week in
the Bahia cocoa zone, alleviating the drought since early
January and improving prospects for the coming temporao,
although normal humidity levels have not been restored,
Comissaria Smith said in its weekly review.
    The dry period means the temporao will be late this year.
    Arrivals for the week ended February 22 were 155,221 bags
of 60 kilos making a cumulative total for the season of 5.93
mln against 5.81 at the same stage last year. Again it seems
that cocoa delivered earlier on consignment was included in the
arrivals figures.
    Comissaria Smith said there is still some doubt as to how
much old crop cocoa is still available as harvesting has
practically come to an end. With total Bahia crop estimates
around 6.4 mln bags and sales standing at almost 6.2 mln there
are a few hundred thousand bags still in the hands of farmers,
middlemen, exporters and processors.
    There are doubts as to how much of this cocoa would be fit
for export as shippers are now experiencing dificulties in
obtaining +Bahia superior+ certificates.
```

```
BankAmerica Corp is not under
pressure to act quickly on its proposed equity offering and
would do well to delay it because of the stock's recent poor
performance, banking analysts said.
    Some analysts said they have recommended BankAmerica delay
its up to one-billion-dlr equity offering, which has yet to be
approved by the Securities and Exchange Commission.
    BankAmerica stock fell this week, along with other banking
issues, on the news that Brazil has suspended interest payments
on a large portion of its foreign debt.
    The stock traded around 12, down 1/8, this afternoon,
after falling to 11-1/2 earlier this week on the news.
    Banking analysts said that with the immediate threat of the
First Interstate Bancorp &lt;I> takeover bid gone, BankAmerica is
under no pressure to sell the securities into a market that
will be nervous on bank stocks in the near term.
    BankAmerica filed the offer on January 26. It was seen as
one of the major factors leading the First Interstate
withdrawing its takeover bid on February 9.
    A BankAmerica spokesman said SEC approval is taking longer
```

Words/topics that differentiate the left text from the right one.
Computed with: LDA

```
topic 1: new, york, time, sale, cocoa, mln, smith, crop
```

Figure 6: Result GUI LDA


TextRank differentiation method: simple differentiation

```
Showers continued throughout the week in
the Bahia cocoa zone, alleviating the drought since early
January and improving prospects for the coming temporao,
although normal humidity levels have not been restored,
Comissaria Smith said in its weekly review.
    The dry period means the temporao will be late this year.
    Arrivals for the week ended February 22 were 155,221 bags
of 60 kilos making a cumulative total for the season of 5.93
mln against 5.81 at the same stage last year. Again it seems
that cocoa delivered earlier on consignment was included in the
arrivals figures.
    Comissaria Smith said there is still some doubt as to how
much old crop cocoa is still available as harvesting has
practically come to an end. With total Bahia crop estimates
around 6.4 mln bags and sales standing at almost 6.2 mln there
are a few hundred thousand bags still in the hands of farmers,
middlemen, exporters and processors.
    There are doubts as to how much of this cocoa would be fit
for export as shippers are now experiencing dificulties in
obtaining +Bahia superior+ certificates.
```

```
BankAmerica Corp is not under
pressure to act quickly on its proposed equity offering and
would do well to delay it because of the stock's recent poor
performance, banking analysts said.
    Some analysts said they have recommended BankAmerica delay
its up to one-billion-dlr equity offering, which has yet to be
approved by the Securities and Exchange Commission.
    BankAmerica stock fell this week, along with other banking
issues, on the news that Brazil has suspended interest payments
on a large portion of its foreign debt.
    The stock traded around 12, down 1/8, this afternoon,
after falling to 11-1/2 earlier this week on the news.
    Banking analysts said that with the immediate threat of the
First Interstate Bancorp &lt;I> takeover bid gone, BankAmerica is
under no pressure to sell the securities into a market that
will be nervous on bank stocks in the near term.
    BankAmerica filed the offer on January 26. It was seen as
one of the major factors leading the First Interstate
withdrawing its takeover bid on February 9.
    A BankAmerica spokesman said SEC approval is taking longer
```

Words/topics that differentiate the left text from the right one.
Computed with: TextRank
differentiation method: simple differentiation

```
sale
crop
bahia
cocoa
comissaria
exporter
end
february
bag
kilo
bean
dec
export
shipper
mln
```

Figure 7: Result GUI TextRank

5. Evaluation

In this chapter the results of the implemented approaches of this thesis are evaluated.
To test the quantity of the results of the approaches an automated evaluation is carried out. In this automated evaluation the processing time, the saved reading time and the proportion of retrieved nouns, verbs, adjectives and adverbs are analysed.
To complete the evaluation, the quality of the approaches must be measured, which is done through a human evaluation. In this evaluation, human users are asked to rank the approaches.
Both evaluations work with 100 texts from the Reuters-21578 corpus. 50 of these texts are derived from the category "acq" because this is one of the largest categorise in the corpus. "acq" stands for "mergers/acquisitions" and contains news texts about corporate acquisitions and corporate merges. The other 50 are randomly selected from the entire corpus. This partition of the sample texts ensures that at first texts from the same category with similar content are evaluated. The random texts from the entire corpus make an evaluation of diverse texts possible.
Table 1 displays how many characters the texts have. Figure 8 displays the distribution of text length over all texts. Most texts have between 149 and about 3000 characters with overall six outliers. The most extreme outlier occurs in the random texts and has 14060 characters. On average, texts from "acq" are shorter than texts from the random sample.

	minimum characters	average characters	maximum characters
text length all	149	1303	14060
text length "acq"	149	1064	4213
text length rest	171	1542	14060

Table 1: Number of characters in the texts of the evaluation corpus.

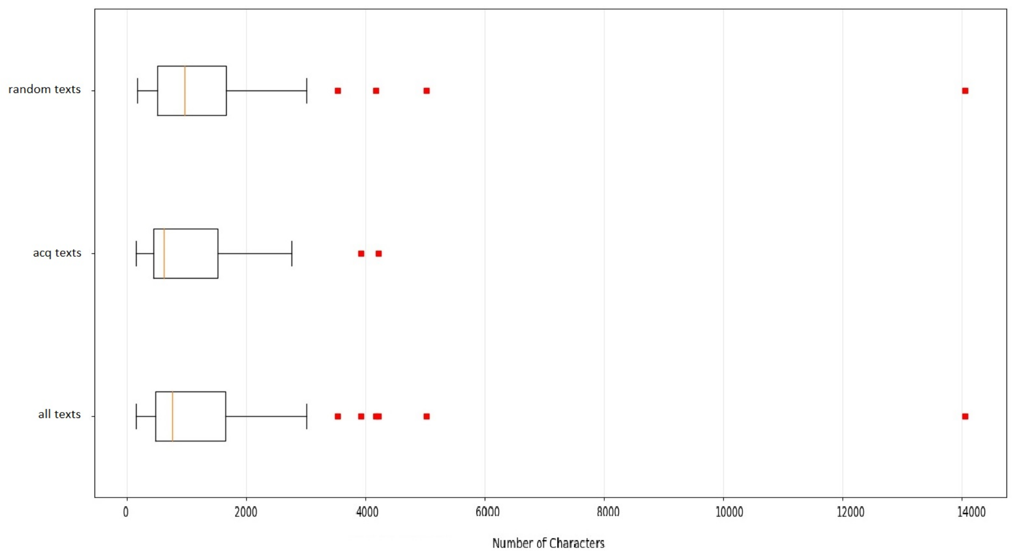


Figure 8: Distribution of the number of characters in the evaluation corpus and partitioned in "acq" texts and the random chosen texts.

5.1. Automated Evaluation

The automated evaluation concentrates on the quantitative aspects of the approaches. To evaluate the quantitative benefit, the processing time, the saved reading time and the proportion of the retrieves nouns, verbs, adjectives and adverbs is tested for every approach and its comparative approaches.
The evaluation also experimentally determines a suitable parameter for LDA.

5.1.1. Setup

The following benchmarks are evaluated for the tf-idf with the simple differentiation and the semantic differentiation, the TextRank with the simple differentiation and the semantic differentiation and the LDA, which does not require differentiation.

- Processing time
  The approaches run on all sample texts. The fastest, average and slowest processing time per approach with its comparative approaches is analysed.
- Saved reading time
  This benchmark estimates how many minutes can be saved by reading the result of the approaches instead of the source text (assuming an average reading speed of 220 words per minute). The minimum, average and maximum of a computation with all sample texts is used.

- Proportion of retrieved nouns, verbs, adjectives and adverbs
  For all sample texts, the average percentage of retrieved nouns, verbs, adjectives and adverbs is computed.

The processing time of the approaches determines the response time and thus the user experience. 0.1 seconds is the threshold below which a user feels that a system reacts instantaneously. 1.0 seconds is the limit for no interruption of the flow of the user and 10 seconds is the limit to keep the attention of the user [Nie94].
Therefore the processing time should be as low as possible. The first benchmark measures the processing time for all approaches.
A study of Hans-Werner Hunziker in 2005 concluded, that the average reading speed of an average reader is 200 to 240 words per minute [Hun06].
For this thesis, the average reading speed is assumed as 220 words per minute. The reduction of the reading time is a feature that contributes to the usability and thus is measured in the second benchmark.
Martin Wachtel describes in his book "Grammatik und vieles mehr" that nouns are necessary for a meaningful conversation because they name facts. Adjectives and adverbs on the other hand only add to a communicative effect. They do not add much meaning to a sentence. The meaning value of verbs is between nouns and adjectives/adverbs [Wac05].
It could be assumed that nouns can summarise a text best, followed by verbs. Adjectives and adverbs describe a text worst; they deliver the least meaning. Because of this assumption the ratio of retrieved nouns, verbs, adjectives and adverbs is measured in the last benchmark.

The following parameters are used for the automated evaluation:

- tf-idf, simple differentiation:
  ratio of best words: 0.33
- tf-idf, semantic differentiation:
  ratio of best words: 0.33
- TextRank, simple differentiation:
  ratio of nodes: 0.33
- TextRank, simple differentiation:
  ratio of nodes: 0.33
- LDA:
  The optimal parameter is experimentally defined in the chapter 5.1.2.

The parameters for tf-idf and TextRank are derived from the TextRank paper [MT04]. The LDA paper does not define a default parameter for the computation [BNJ02] so the default parameter is defined in the result sub-chapter, chapter 5.1.2.

All benchmarks use the sample texts defined above.

5.1.2. Results

The first part of this sub chapter consists of the computation of the default parameter for the LDA. Afterwards the results of the benchmarks processing time, saved reading time and ratio of retrieved nouns, verbs, adjectives and adverbs are discussed.

LDA Default Parameter

As mentioned in the concept chapter the best default values for the tf-idf and the TextRank are a third of the unique words in a text/nodes in the text graph. This number is used in the in all tests below.

In a paper about LDA Blei, Ng and Jordan mention the perplexity of a model as standard measure to discover the best number of topics (n) for the LDA model [BNJ03]. The perplexity is the logarithm of the familiar entropy [JMBB77] or in other words it measures how good a probability distribution or probability model predicts a sample.

Figure 9 shows how the model perplexity and the per word perplexity changes with increasing topics. The curves mirror each other with noise peaks at the same spots. Both curves show no "elbow" from which a best number of topics could be determined. Therefore the best number of topics is chosen by trial and error.

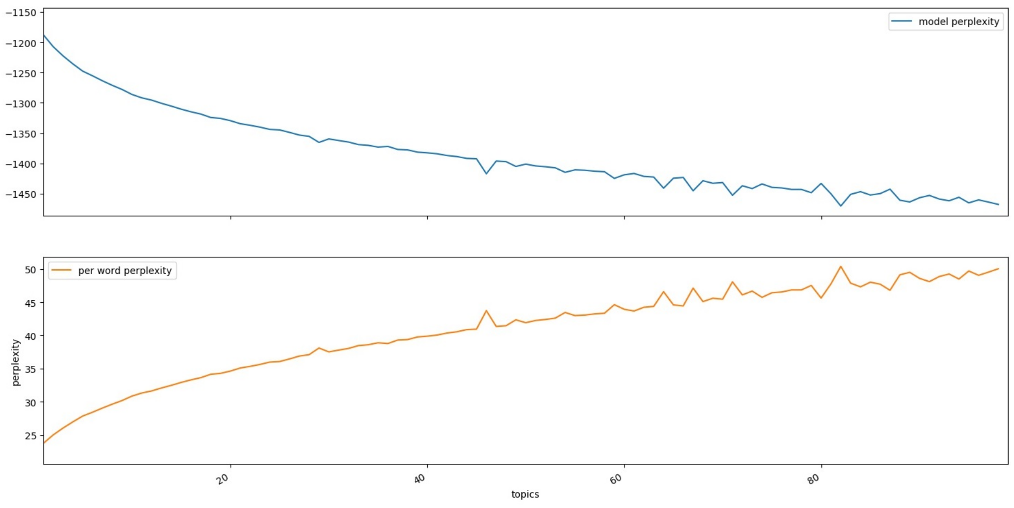


Figure 9: Plotted per word and model perplexity of the LDA for one to 100 topics.

The first attempt to determine n is to use two texts with an average number of character of the sample texts.

Figure 10 shows the mean topic output for one to ten topics.

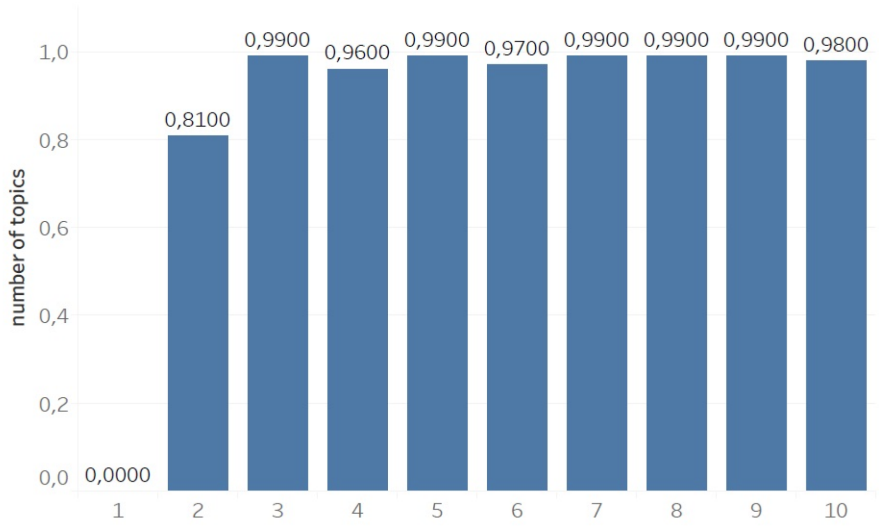


Figure 10: Mean topic assignment for texts with an average length for one to ten generated topics.

From this distribution one clear conclusion can be derived: one topic is never a good choice. All other topic numbers have a similar distribution, so no final statement can be made about the best n.
To provide a better understanding of the resulting topic assignments the results are printed in sub chapter A.3. But even this does not lead to a clear result.

That is why in a second approach the smallest texts of the sample texts are combined and for these from one to ten topics are computed. Figure 11 shows that with the minimal text length of about 200 characters at least six topics are needed to get a result.

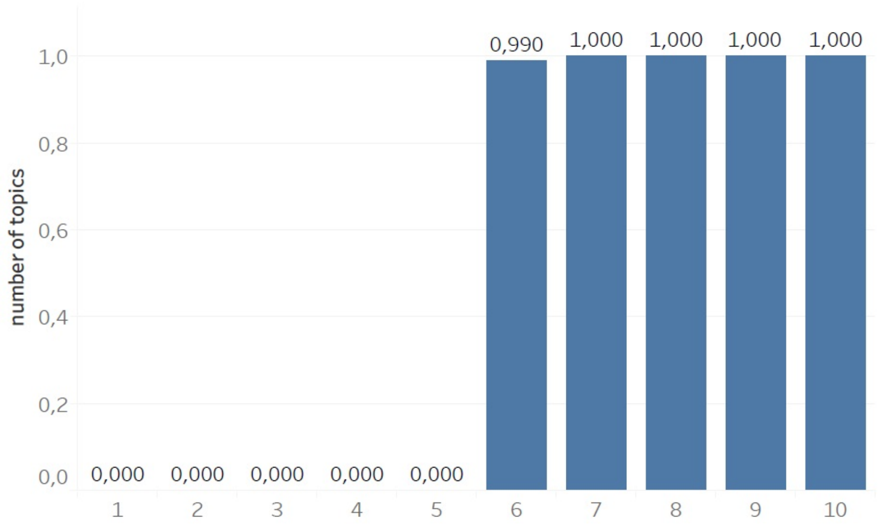


Figure 11: Mean topic assignment for texts with a small length for one to ten generated topics.

But most texts have many more characters than the smallest one. Because of this, in a third approach the smallest text of the sample texts is combined with an average text of the sample texts to determine the best n. Figure 12 clearly shows, that five topics is the best n even for small texts. Above five the probabilities of the computed topics get too small and the assignment rate drops.

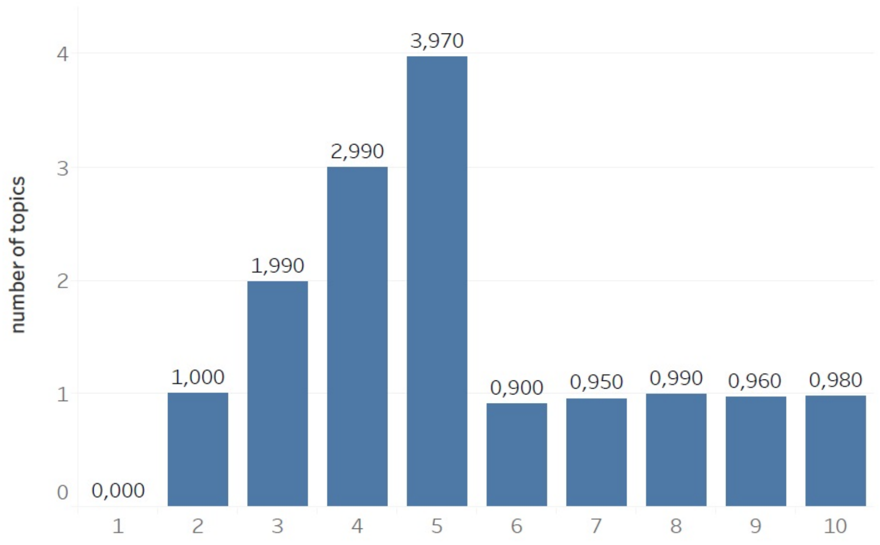


Figure 12: Mean topic assignment for a text with a small and an average text length for one to ten generated topics.

The possibility that two very small texts with about 200 characters are combined in the evaluation is very low. Therefore it should not be a problem to declare five as default n for this evaluation.
If small texts should be computed the default n has to be increased to six.

Benchmark Results

Table 2 shows an overview of the results of the automated evaluation. Processing time is measured in seconds, the saved reading time in minutes and the retrieved word types in percent. The best values are mark with a bold font.

	LDA	$TextRank_{simple}$	$TextRank_{semantic}$	$tf\text{-}idf_{simple}$	$tf\text{-}idf_{semantic}$
processing time (sec)	0.36 ± 035	0.2 ± 0.11	0.27 ± 0.17	**0.15** ± 0.09	0.22 ± 0.15
saved reading time (min)	**0.86** ± 0.72	0.81 ± 0.73	0.81 ± 0.73	0.79 ± 0.73	0.79 ± 0.73
summary size (number of words)	**7.96** ± 3.60	18.84 ± 14.17	18.88 ± 14.11	22.52 ± 15.55	21.78 ± 15.20
retrieved word types					
noun (%)	70.70 ± 14.28	**74.97** ± 14.37	74.06 ± 14.74	68.57 ± 13.22	67.75 ± 12.94
verb (%)	**13.18** ± 5.86	12.69 ± 5.88	12.46 ± 5.08	12.57 ± 6.22	12.23 ± 5.93
adjective (%)	21.81 ± 8.39	**17.42** ± 7.86	17.93 ± 7.74	21.25 ± 14.28	21.5 ± 14.08
adverb (%)	15.16 ± 4.87	**5.46** ± 2.98	6.23 ± 3.94	7.49 ± 3.17	7.98 ± 3.84

Table 2: Comparison of the algorithms. Showing mean and standard deviation. The best values are mark with a bold font.

Runtime speed
The test is executed on the following configuration:

- Intel Core i5-6200 CPU @2.30 GHz
- 8 GB RAM
- Windows 10 (64 Bit)
- Python 3.6

Furthermore an elaborate IDE and other background tasks have influence on the processing time.

Figure 13 shows the processing times of the different approaches with standard parameters. The maximum time of the LDA is exorbitantly higher than its average time. This peak only occurs when the approach is started for the first time. When it is used continuously the runtime decreases dramatically.
Overall the fastest approach is the tf-idf with simple differentiation. The LDA has the overall worst processing time.

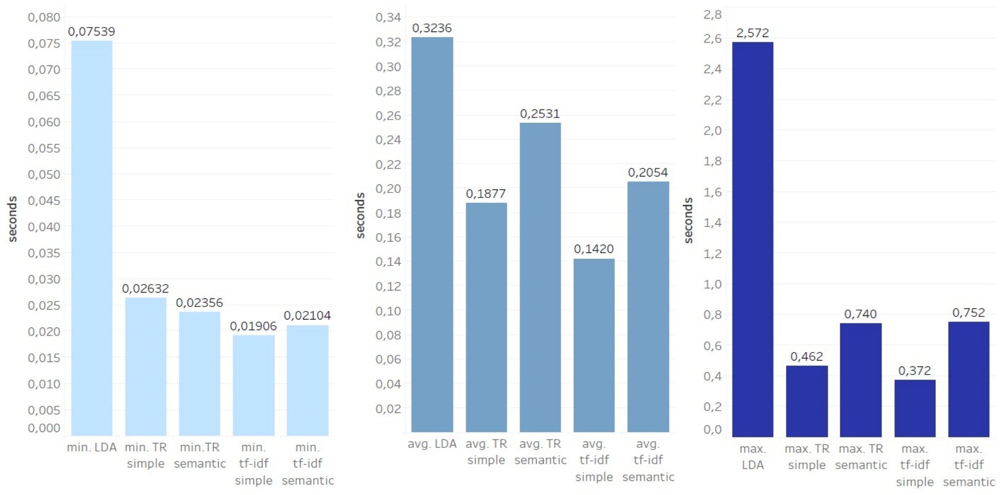


Figure 13: Minimal, mean and maximal processing time of all approaches

Saved reading time

The saved reading time benchmark is computed by running all approaches through all sample texts with the standard parameters.

Figure 14 shows that the saved reading time is similar over all approaches. But with the LDA most time (68.93%) can be saved on average. The least time is saved with tf-idf on average.

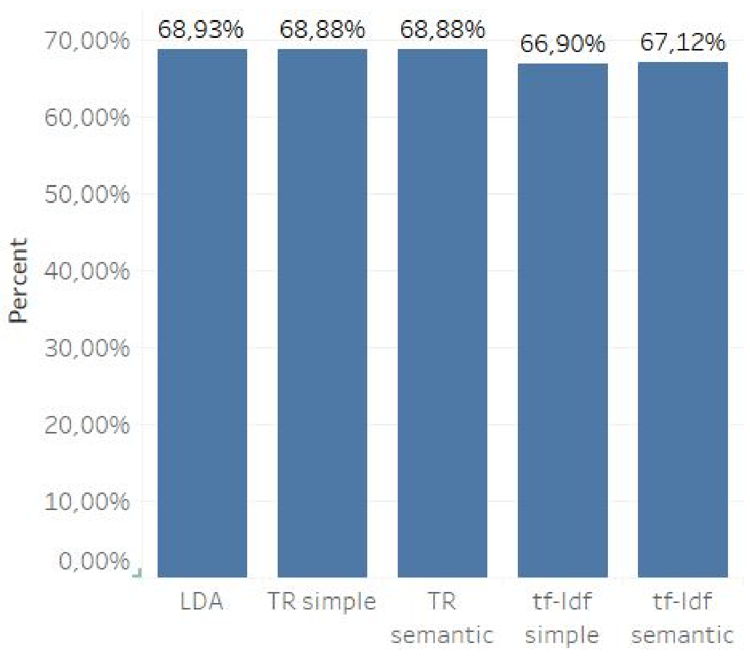


Figure 14: Mean saved reading time in contrast to reading the whole source text.

Proportion of Retrieved Nouns, Verbs, Adjectives and Adverbs

Figure 15 shows the ratio of retrieved words for each type per text, in relation to the total word count of a text. The percentages do not add up to 100% because it is a mean value for every word type.

All approaches retrieve mostly nouns. The second most retrieved word type is adjectives. This is interesting because verbs should carry more meaning.

TextRank with simple differentiation retrieves the most nouns and a fair amount of verbs and the least adjectives and adverbs on average. Even in the worst case it retrieves the most nouns. The LDA retrieves more verbs on average and in the worst case. In the best case the tf-idf with simple differentiation retrieves the most verbs. TextRank with simple differentiation retrieves the least adjectives and adverbs in all cases.

The adjective/adverb retrieval rate combined with the noun retrieval rate makes TextRank with simple differentiation the best approach in terms of word type retrieval.

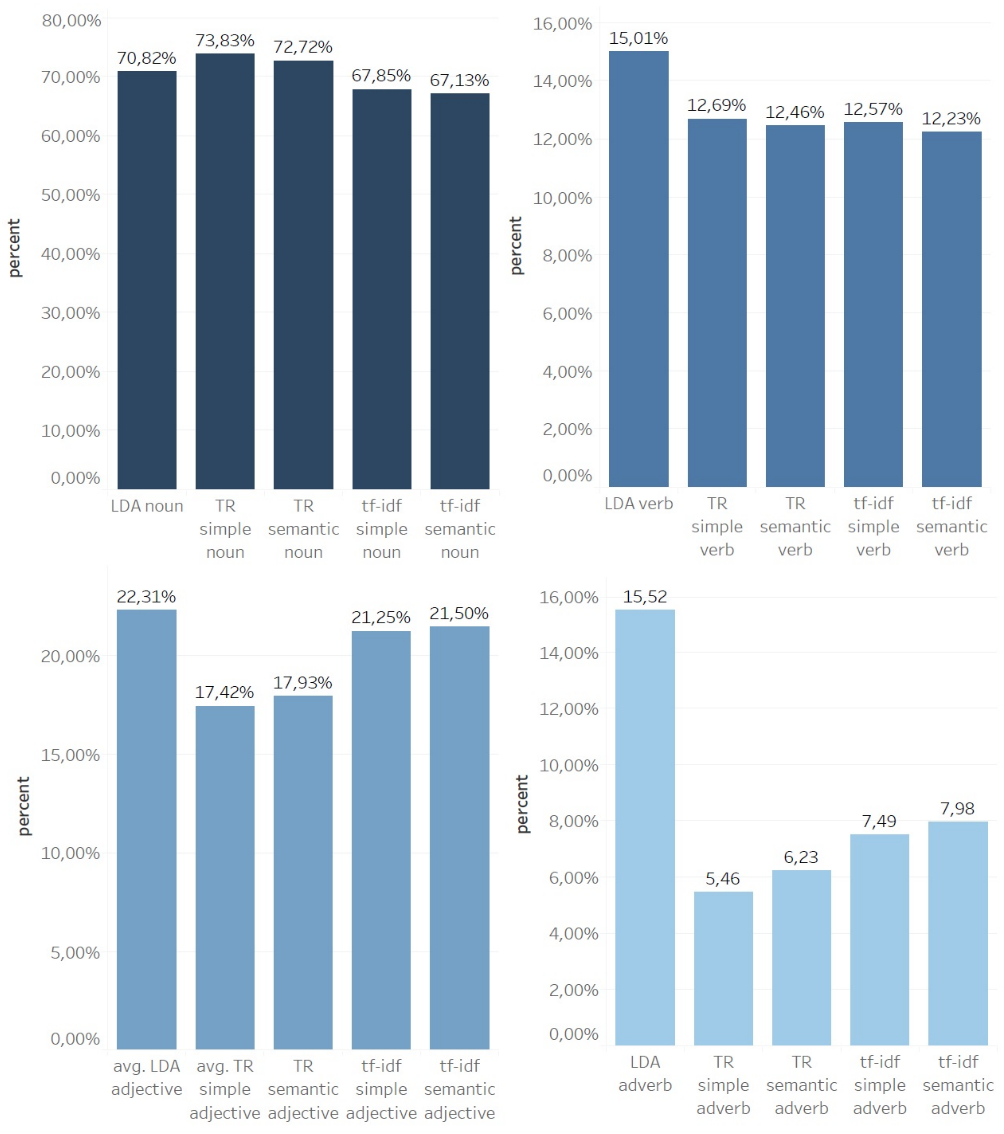


Figure 15: Mean percentage of the retrieved word types.

5.1.3. Discussion

The automatic evaluation does not lead to an overall best approach. Every approach excels in one benchmark. In the processing time the tf-idf with simple differentiation has the best values. The most reading time can be saved with the LDA and the TextRank with simple differentiation retrieves the most meaningful word types. The only thing that can be stated is that the semantic differentiation approach never excels, neither for tf-idf nor for TextRank.

5.2. Human Evaluation

The human evaluation focuses on the qualitative parts of the evaluation. It consists of two sub evaluations: the first provided valuable insights on how to conduct an evaluation. Also some insights concerning the LDA were discovered. The second one builds on these insights to compare the quality of the approaches. Moreover valuable feedback for the approaches was given.
In the adjusted evaluation the display and the questions of the evaluation were clarified. Furthermore the feedback collected in the initial evaluation is incorporated in the approaches.

5.2.1. Initial Evaluation

In the initial evaluation the participants should use a GUI to answer questions about the results of the approaches. At the end the users should rate all approaches from one to five.

Setup
The following insights per approach should be derived from the evaluation:

1. How well does an approach describe the text overall?
2. How many words were considered a bad description?
3. Does the result only contain the differences of the source texts?
4. How do the testers rank the approaches?

These questions should reveal what the users think about the quality of the approaches. They should assess the results of the approaches as a whole and not concentrate on single words. As non-native speakers it is not possible to make an informed decision on single words. The first insight should measure the quality of the description of the concepts an approach retrieves from the source text. The second one measures how many words retrieved by a particular approach are not suitable. Even if an approach is considered not bad overall, the second measure of unsuitable words gives a better insight if an approach is not bad, good or really good. With the third insight the main task of the approaches is evaluated, whether only differences are retrieved or not. In the end the fourth question is asked after all approaches are judged twice. Now the participants can make an educated ranking of the approaches because they saw the results of every approach twice.

The evaluation GUI consists of two parts. Figure 16 shows the first part. In this one the users put in their test person number. They then choose between two text pairs, which can be selected with the buttons "Text One" and "Text Two". Afterwards the users choose between tf-idf, TextRank and LDA. Depending on the approach, a comparative method, simple or semantic differentiation, has to be chosen. The results of the approaches are displayed in a new window.

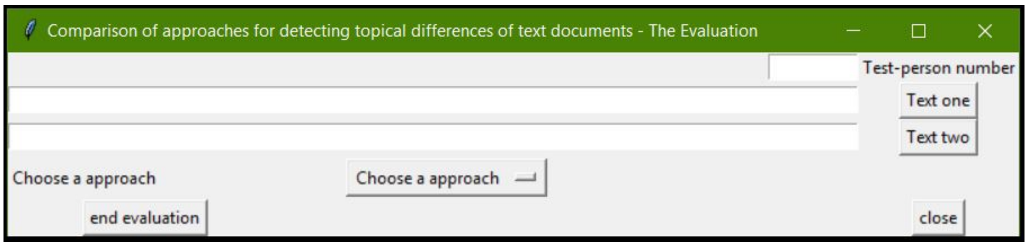
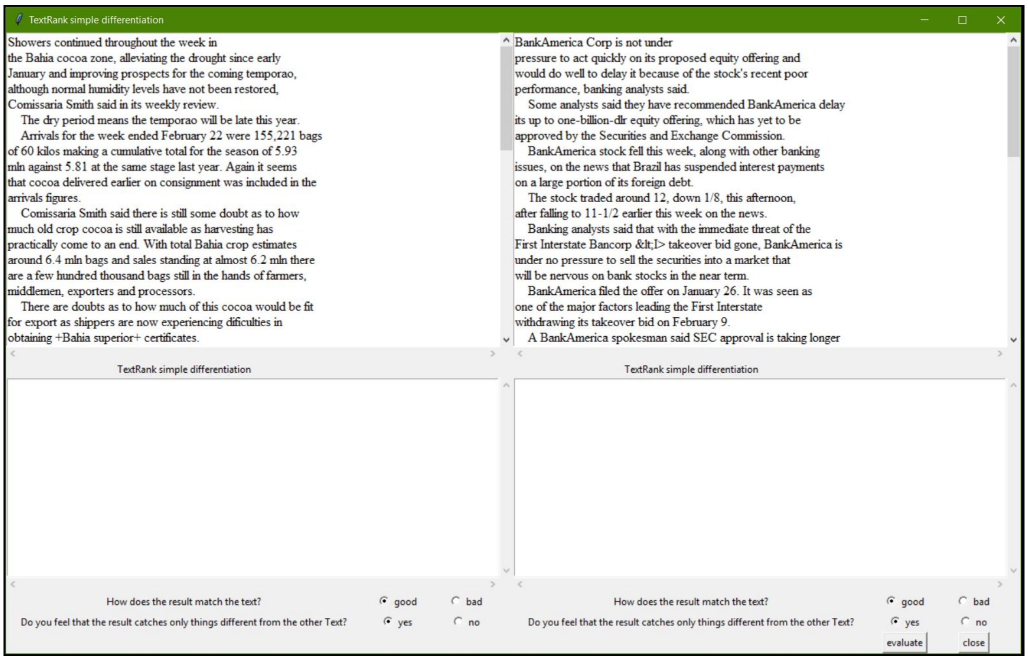

Comparison of approaches for detecting topical differences of text documents - The Evaluation
Test-person number
Text one
Text two
Choose a approach
Choose a approach
end evaluation
close

Figure 16: Initial testing GUI without input.

Every approach is tested with two text pairs from the reut2-000 file of the Reuters-21578 corpus:

- id 1 and id 4
- id 107 and id 192

These text pairs were randomly chosen from the corpus. The only constraint was, that both texts have roughly the same size.
In Figure 17 the result of the TextRank computation with simple differentiation is displayed. In this result GUI the users should pick the words that describe the source text worst. The users should additionally determine if the whole result matches the text or not and if the result only represents the differences between the two source texts.


TextRank simple differentiation

Showers continued throughout the week in
the Bahia cocoa zone, alleviating the drought since early
January and improving prospects for the coming temporao,
although normal humidity levels have not been restored,
Comissaria Smith said in its weekly review.
The dry period means the temporao will be late this year.
Arrivals for the week ended February 22 were 155,221 bags
of 60 kilos making a cumulative total for the season of 5.93
mln against 5.81 at the same stage last year. Again it seems
that cocoa delivered earlier on consignment was included in the
arrivals figures.
Comissaria Smith said there is still some doubt as to how
much old crop cocoa is still available as harvesting has
practically come to an end. With total Bahia crop estimates
around 6.4 mln bags and sales standing at almost 6.2 mln there
are a few hundred thousand bags still in the hands of farmers,
middlemen, exporters and processors.
There are doubts as to how much of this cocoa would be fit
for export as shippers are now experiencing dificulties in
obtaining +Bahia superior+ certificates.

BankAmerica Corp is not under
pressure to act quickly on its proposed equity offering and
would do well to delay it because of the stock's recent poor
performance, banking analysts said.
Some analysts said they have recommended BankAmerica delay
its up to one-billion-dlr equity offering, which has yet to be
approved by the Securities and Exchange Commission.
BankAmerica stock fell this week, along with other banking
issues, on the news that Brazil has suspended interest payments
on a large portion of its foreign debt.
The stock traded around 12, down 1/8, this afternoon,
after falling to 11-1/2 earlier this week on the news.
Banking analysts said that with the immediate threat of the
First Interstate Bancorp <I> takeover bid gone, BankAmerica is
under no pressure to sell the securities into a market that
will be nervous on bank stocks in the near term.
BankAmerica filed the offer on January 26. It was seen as
one of the major factors leading the First Interstate
withdrawing its takeover bid on February 9.
A BankAmerica spokesman said SEC approval is taking longer

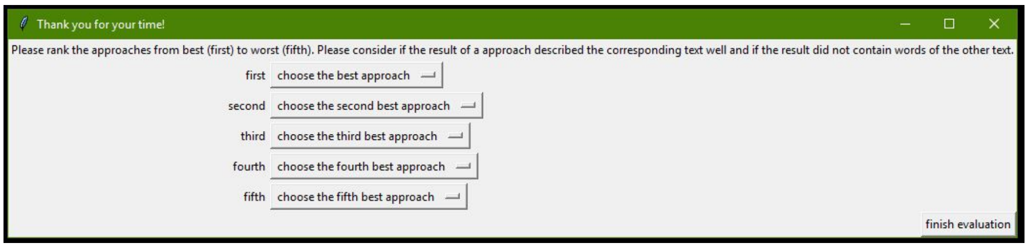
TextRank simple differentiation | TextRank simple differentiation

How does the result match the text? [✓ good] bad
Do you feel that the result catches only things different from the other Text? [✓ yes] no

How does the result match the text? [✓ good] bad
Do you feel that the result catches only things different from the other Text? [✓ yes] no

Figure 17: Result of the TextRank computation with simple differentiation, displayed in the testing GUI.

The users are asked to rate all three approaches with the two comparative approaches with two different text pairs. The evaluation ends with an overall ranking of all approaches from one to five as demonstrated in figure 18.


Thank you for your time!

Please rank the approaches from best (first) to worst (fifth). Please consider if the result of a approach described the corresponding text well and if the result did not contain words of the other text.

first	choose the best approach
second	choose the second best approach
third	choose the third best approach
fourth	choose the fourth best approach
fifth	choose the fifth best approach

Figure 18: Last window of the evaluation in which the approaches should be ranked from one to five.

After the users judged everything in the result GUI they click the evaluate button and the evaluation is saved in a JSON format. The following data are saved:

- test person
  ID for the participant who conducts the evaluation.
- text one
  Path of the first text
- text two
  path of the second text
- result one
  Result of the computation for the first text.
- result two
  Result of the computation for the second text.
- approach
  Approach that is used for the computation.
- Differentiation method
  If TextRank or tf-idf is used the differentiation method, otherwise an empty string is saved.
- How well result one describes the source text one:
  Either "good" or "bad" as chosen by the participant.
- How well result two describes the source text two:
  Either "good" or "bad" as chosen by the participant.
- Bad words for text one
  The participant can mark words they think are unsuitable.
- Bad words for text two
  The participant can mark words they think are unsuitable.
- Does the first result only contain differences from text one to text two?
  Either "yes" or "no" as chosen by the participant.
- Does the second result only contain differences from text two to text one?
  Either "yes" or "no" as chosen by the participant.

The ranking of the approaches is also saved in a JSON format. The following data is saved:

- test person
  ID for the participant who conducts the evaluation.
- Ranking for each approach

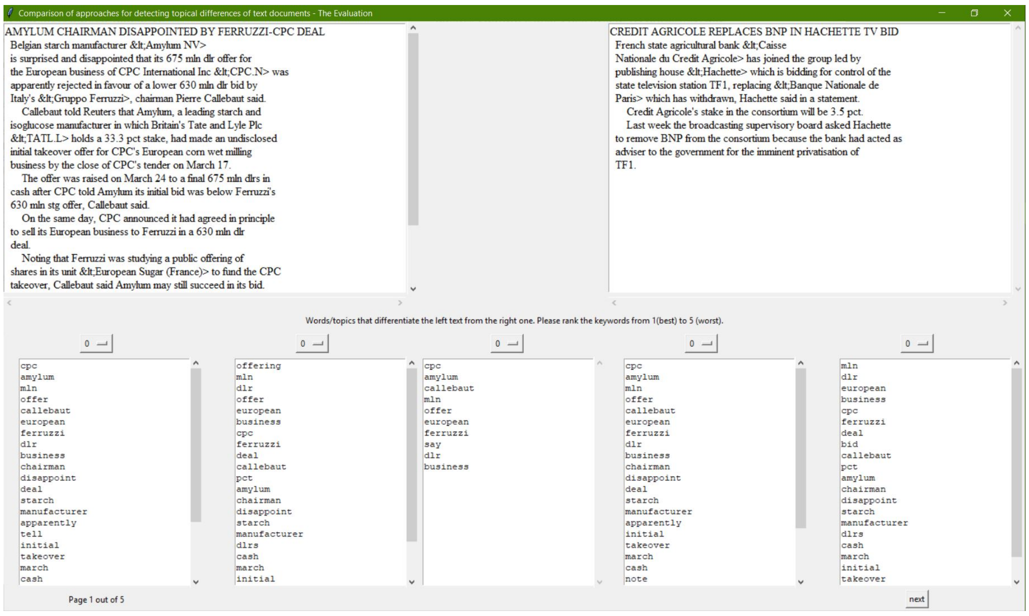
Results and Discussion
The evaluation was only conducted with seven participants. Therefore the obtained results are not meaningful.
The participants provided many useful remarks. These included, that the display of the results was not intuitive. The two results for both texts confused the users. Furthermore the questions asked lead to a wrong impression about the thesis. The focus was too heavy on how well the results describe the texts. The main objective of the difference retrieval was not recognized. Finally the result of the LDA computation could be increased by not only deleting topics that represents both texts, but by removing words from the topics that occur in the topics of both texts.
This feedback is incorporated into the thesis and the adjusted evaluation.

5.2.2. Adjusted Evaluation

The feedback gathered in the initial evaluation is worked into the thesis and the adjusted evaluation. Before the evaluation is given to the users a pretest examined whether the evaluation is intuitive and asks the right questions or not. In this pretest a user with no prior knowledge is asked to do the evaluation and to give feedback about it. After the evaluation ends the user is asked whether they understand the intention of the evaluation. After a positive answer to this question the evaluation is carried out.

Setup
The main point of the adjusted evaluation is to rank all approaches from one to five for all sample texts. This ranking gives an overview about the quality of the result of the approaches. Ten users execute the evaluation. Every user rates different texts. With this approach it is possible to rank a wide range of texts. A ground assumption is that every user would rate the other texts in the same way they rated their five text pairs. It would not be possible to ask every user to rate 100 texts. Furthermore would it be difficult to get a broader audience to rate the sampled texts for more than once. This would be beyond the scope of a bachelor thesis.
As in the initial evaluation, the users judge with a GUI. Figure 19 displays the main evaluation GUI. In this one the two source texts are shown. Beneath them the results of all approaches are displayed side by side. The results contain the differences between the left and the right text. All results should be ranked from one to five, where one is the best and five the worst ranking. If the users feel that two approaches deserve the same rank they are allowed to rank them equally. Sometime the results for tf-idf with simple and semantic differentiation and for TextRank with simple and semantic differentiation issue the same result.
The source texts are randomly chosen from the 100 sampled texts. The first 50 texts are

from the "acq" category, the remaining ones are the random texts. The results from the approaches are displayed at a random place beneath to reduce the bias of the result that is displayed in the first place. The users are not informed which result is computed with which approach.


Comparison of approaches for detecting topical differences of text documents - The Evaluation

AMYLUM CHAIRMAN DISAPPOINTED BY FERRUZZI-CPC DEAL
Belgian starch manufacturer <Amylum NV>
is surprised and disappointed that its 675 mln dlr offer for
the European business of CPC International Inc <CPC.N> was
apparently rejected in favour of a lower 630 mln dlr bid by
Italy's <Gruppo Ferruzzi>, chairman Pierre Callebaut said.
Callebaut told Reuters that Amylum, a leading starch and
isoglucose manufacturer in which Britain's Tate and Lyle Plc
<TATL.L> holds a 33.3 pct stake, had made an undisclosed
initial takeover offer for CPC's European corn wet milling
business by the close of CPC's tender on March 17.
The offer was raised on March 24 to a final 675 mln dlrs in
cash after CPC told Amylum its initial bid was below Ferruzzi's
630 mln stg offer, Callebaut said.
On the same day, CPC announced it had agreed in principle
to sell its European business to Ferruzzi in a 630 mln dlr
deal.
Noting that Ferruzzi was studying a public offering of
shares in its unit <European Sugar (France)> to fund the CPC
takeover, Callebaut said Amylum may still succeed in its bid.

CREDIT AGRICOLE REPLACES BNP IN HACHETTE TV BID
French state agricultural bank <Caisse
Nationale du Credit Agricole> has joined the group led by
publishing house <Hachette> which is bidding for control of the
state television station TF1, replacing <Banque Nationale de
Paris> which has withdrawn, Hachette said in a statement.
Credit Agricole's stake in the consortium will be 3.5 pct.
Last week the broadcasting supervisory board asked Hachette
to remove BNP from the consortium because the bank had acted as
adviser to the government for the imminent privatisation of
TF1.

Words/topics that differentiate the left text from the right one. Please rank the keywords from 1(best) to 5 (worst).

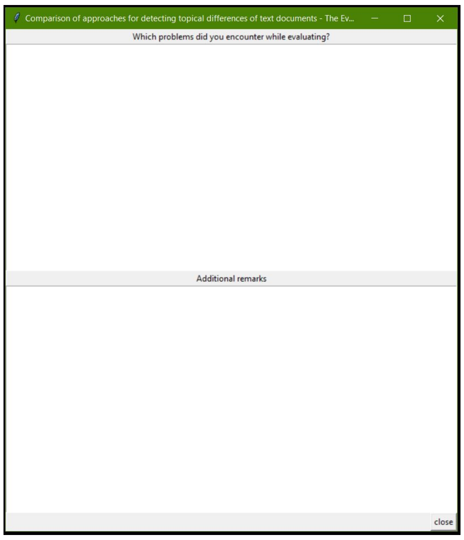
```
0             0             0             0             0
cpc           offering      cpc           cpc           mln
amylum        mln           amylum        amylum        dlr
mln           dlr           callebaut     mln           european
offer         offer         mln           offer         business
callebaut     european      offer         callebaut     cpc
european      business      european      european      ferruzzi
ferruzzi      cpc           ferruzzi      ferruzzi      deal
dlr           ferruzzi      say           dlr           bid
business      deal          dlr           business      callebaut
chairman      callebaut     business      chairman      pct
disappoint    pct                         disappoint    amylum
deal          amylum                      deal          chairman
starch        chairman                    starch        disappoint
manufacturer  disappoint                  manufacturer  starch
apparently    starch                      apparently    manufacturer
tell          manufacturer                initial       dlrs
initial       dlrs                        takeover      cash
takeover      cash                        march         march
march         march                       cash          initial
cash          initial                     note          takeover
```

Page 1 out of 5

next

Figure 19: GUI for the adjusted human evaluation. Here all results should be ranked from one to five.

After five pages of evaluation the window displayed in figure 20 opens. In this one the users can comment if they had any problems while evaluating and leave remarks.


Comparison of approaches for detecting topical differences of text documents - The Ev...

Which problems did you encounter while evaluating?

Additional remarks

close

Figure 20: Last page of the human evaluation in which the users can leave remarks about difficulties they encountered and everything else they wish to share.

Results
Tf-idf with simple differentiation is rated best. In contrast to this LDA is rated the worst. Furthermore the simple differentiation is always better rated than the semantic differentiation. This is shown in figure 21. In A.4 on page 65 the remarks the users made in the evaluation can be found. The most difficulties occur because the texts are difficult to read and understand and because long texts lead to long word lists. Some users are confused that some approaches resulted in the same word lists. A common remark is that the results of the approaches often contain words with no or little meaning. Furthermore participants suggests adaptations of the display of the results. Some users add verbally that the approaches do not describe the texts well and that not all, or not even all important differences are retrieved. In other words the users are not satisfied with the results of any of the approaches. Overall the users remark verbally that their concentration dropped over time. To include this remark in the evaluation a second result of the evaluation is also shown in figure 21. Each second bar belongs to the evaluation result without the last rating of every participant. In this result the last rating per user is not considered. This figure demonstrates that even if the concentration of the users decreased while evaluating their rating of the approaches did not change.

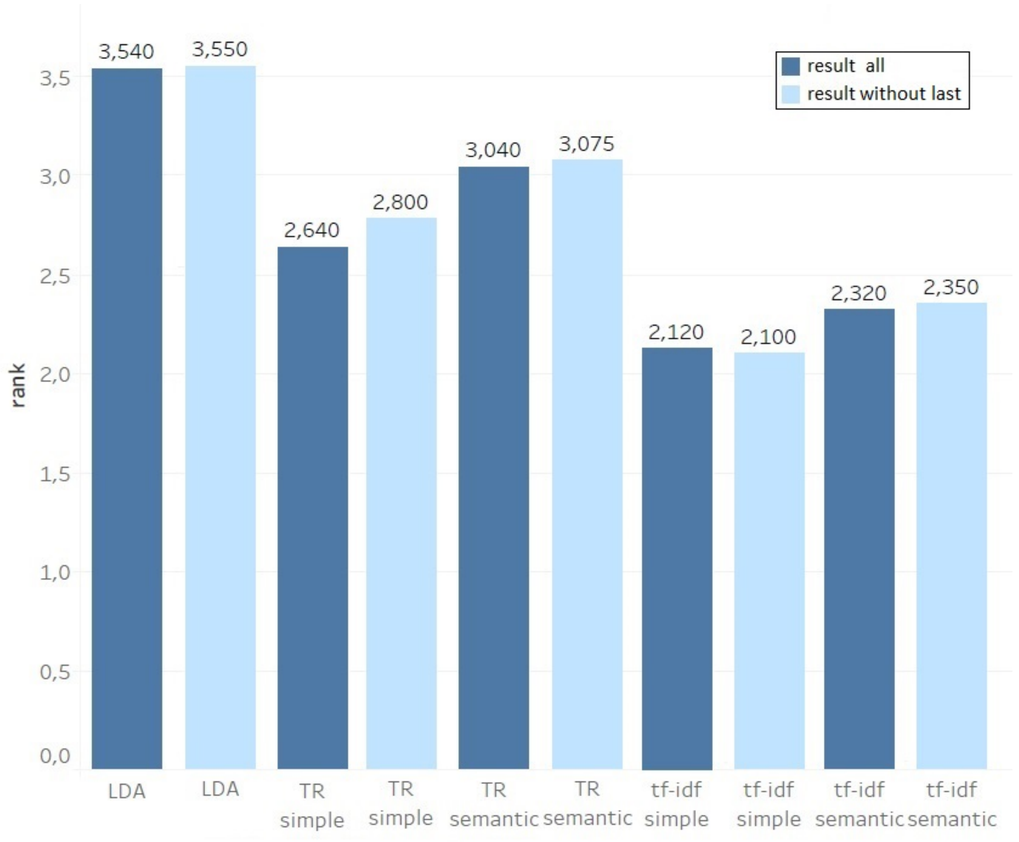


Figure 21: Users' ranking of the result quantity of the approaches (lower is better).

Discussion

The results of the evaluation shown in figure 21 indicate that no approach was good enough to mostly get a first ranking. The remarks in A.4 on page 65 also indicate that the quality of the approaches could be increased.
Moreover the constant better rating for the simple differentiation in contrast to the semantic differentiation leads to the insight that removing synonyms, hypernyms and hyponyms is a qualitatively worse approach than removing equal words.
Overall the baseline of this thesis, the tf-idf outperformed the more complex approaches in terms of quality.

5.3. Discussion Of The Evaluation

Considering both the automated and the human evaluation the tf-idf with the simple differentiation is the fastest approach whose results have the best quality. But all approaches still need to be refined to get a qualitatively better outcome. Perhaps the underlying approaches should be changed or the way of difference extraction should be altered. Moreover the display of the results could be adapted.

6. Conclusion

This chapter concludes this thesis. First all developed approaches and their evaluation are recapitulated. Furthermore the implications and limitations of this thesis are discussed. Finally ideas for future work and further research are reviewed.

6.1. Summary

This thesis addressed the problem of difference extraction from text documents. Research in the field of text mining and information retrieval is only concerned with the similarities between texts or the summarisation of them. Therefore new approaches had to be developed.
Approaches for feature representation and text summarisation are well researched and could be extended to differentiations approaches. In particular this thesis extended the following algorithms from these fields: tf-idf, TextRank and LDA.
A pipeline for the difference extraction was implemented. First two text were preprocessed, then the most important concepts of the texts were extracted using the algorithm above. Afterwards the differences between the most important text concepts were extracted with the extension of the algorithms. The extension compared the concepts and removed all similarities. For tf-idf and TextRank two different filtering methods were implemented. Firstly a simple differentiation method that only filtered the words that occur in both concept lists. The LDA used the same principle for the difference extraction. And secondly a semantic differentiation extraction method that filtered all words that were in a synonym, hypernym or hyponym relationship. In the end the remaining concepts were displayed.
The extraction of the most interesting concepts with the existing algorithms had a good performance. Two evaluation participants made remarks concerning the retrieved concepts. One participant criticised that sometimes "keywords were not meaningful (e.g. may, also)". The other participant criticised that compound words were split, this was due to the chosen method of preprocessing. The other participants were satisfied with the retrieved concepts.
The extended approaches indeed only retrieved differences. In A.1.5 on page 55 the results of the difference extraction for all approaches with both difference extraction methods could be seen. Though these differences were not very descriptive of the source texts. It is quite difficult to understand the results as the differences between the first and the second text without any explanation. Keywords alone seemed not descriptive enough to represent the complex concept of differences between texts. But these results come only from a limited amount of participants. With more participants the results could have

differed. Moreover the evaluation participants were non English native speakers that participated in an evaluation on an English corpus. If they had conducted the evaluation on a corpus in their native language the results could have been different. The statements above can only be hypotheses made under the assumptions of the limited evaluation. The number of the evaluated texts should have been big enough to be empirically relevant. A bigger sample size would have lead to better results. Increasing the number of evaluated texts or of participants was not possible due of the time and resource constraints of a bachelor thesis.
Qualitatively all implemented approaches worked quite well in the benchmark tests in chapter 5.1. The processing times of all approaches were fast enough to not interfere with the flow of the user. But they were not fast enough that the user would have a feeling of an instantly reacting system. This is bad, because the approaches will not stand alone. The approaches will be integrated in other systems like the GUI the results were shown in. These systems add to the processing time. Furthermore all approaches led to a smaller reading time in contrast to reading the full source text. And finally all approaches retrieved mostly nouns. Nouns carry the most meaning of texts therefore if mostly nouns were retrieved the most meaning was conserved. The second most retrieved word type is adjective. These carry less meaning than verbs therefore it would have been better if the approaches retrieved more verbs than adjectives. These benchmarks were only tested for an English corpus. The processing time, the saved reading time and the ratio of retrieved word types can differ in different languages. The approaches were also only tested on texts with a limited length. The performance could have been different on longer texts. On longer texts the processing time could have increased to a level where the flow of the user would have been interrupted.

6.2. Future Work

In future work the difference extraction can be improved in many ways. Some of them are listed below:

1. better description of the source texts
2. clearer difference extraction / more descriptive differences
3. better result display

For a better description of the source texts, different underlying algorithm can be used. For example the paragraph vector introduced by Quoc Le and Tomas Mikolov in the doc2vec implementation could be considered for the VSM [LM14]. The field of topic models could be represented by "hidden topic Markov models" as described by Amit Gruber, Yair Weiss and Michal Rosen-Zvi [GWRZ07]. Instead of using the keyword extraction of TextRank, the betweenness centrality of nodes in a text graph could be used,

as introduced by Marta Gruszecka and Michal Pikusa in their paper about using text network analysis in corpus studies [GP15]. The summarisation technique of the TextRank algorithm could also be explored. In addition new Natural Language Processing fields could be explored for better underlying approaches. The preprocessing could also be revised to address the first remark. Using bi-grams or n-grams to extract compound words would be imaginable.
To improve the difference extraction, the underlying approach of removing all similarities to find the differences should be changed. There may be a better way to obtain differences.
The last remark mainly concerns the display of the results. To address this the GUI implementation could be changed. For example a different GUI library could be used. Tkinter is a very simple and unsophisticated library, others could lead to a more intuitive user interface. The display of the result could also group the resulting concepts and make reading them easier.

References

[Ale14] Nikolaos Aletras. *Interpreting Document Collections with Topic Models*. PhD thesis, Department of Computer Science The University of Sheffield, 09 2014.

[APA$^+$17] Mehdi Allahyari, Seyedamin Pouriyeh, Mehdi Assefi, Saeid Safaei, Elizabeth D Trippe, Juan B Gutierrez, and Krys Kochut. Text summarization techniques: A brief survey. *arXiv preprint arXiv:1707.02268*, 2017.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[BKL14] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly UK Ltd., 2014.

[BL09] David M Blei and John D Lafferty. Topic models. *Text mining: classification, clustering, and applications*, 10(71):34, 2009.

[BNJ02] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Advances in neural information processing systems*, 1:601–608, 2002.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[DM07] Dipanjan Das and André FT Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.

[Fel06] Ronen Feldman. *Text Mining Handbook*. CAMBRIDGE UNIVERSITY PRESS, 2006.

[GP15] Marta Gruszecka and Michal Pikusa. Using text network analysis in corpus studies–a comparative study on the 2010 tu-154 polish air force presidential plane crash newspaper coverage. *International Journal of Social Science and Humanity*, 5(2):233, 2015.

[GWRZ07] Amit Gruber, Yair Weiss, and Michal Rosen-Zvi. Hidden topic markov models. In *Artificial intelligence and statistics*, pages 163–170, 2007.

[Hun06] Hans-Werner Hunziker. Im Auge des Lesers: foveale und periphere Wahrnehmung-vom Buchstabieren zur Lesefreude. *The Eye of the Reader: Foveal and Peripheral Perception-from Letter Recognition to the Joy of Reading). Zurich: Transmedia*, 2006.

[J$^+$11] Anjali Ganesh Jivani et al. A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, 2(6):1930–1938, 2011.

[JMBB77] Fred Jelinek, Robert L Mercer, Lalit R Bahl, and James K Baker. Perplexity—a measure of the difficulty of speech recognition tasks. *The Journal of the Acoustical Society of America*, 62(S1):S63–S63, 1977.

[Liu11] Bing Liu. *Web Data Mining*. Springer-Verlag GmbH, 2011.

[LM14] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[LP12] Elena Lloret and Manuel Palomar. Text summarisation in progress: A literature review. *Artif. Intell. Rev.*, 37(1):1–41, January 2012.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Pr., 2008.

[MT04] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *EMNLP*, volume 4, pages 404–411, 2004.

[Nie94] Jakob Nielsen. *Usability Engineering*. Elsevier LTD, Oxford, 1994.

[Por80] Martin F Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[Sor10] Mohammad S Sorower. A literature survey on algorithms for multi-label learning. *Oregon State University, Corvallis*, 18, 2010.

[TP10] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

[Wac05] Martin Wachtel. *Grammatik und vieles mehr (German Edition)*. Peter Lang GmbH, 2005.

List of Figures

List of Tables

List of Terms

Bag-of-words

Approach that only considers the number of the occurrences of words in a text. Word order and semantics are disregarded.

Bi-gram

A token consisting of two consecutive words from a text. Bi-grams are the smallest n-grams.

Bigram language model

Complex form of a language model.

Classification

Method to label objects with already known labels.

Clustering

Method to sort objects into priorly unknown categories.

Corpus

Collection of texts. On these texts text mining can be performed.

Feature

A measurable property that is characteristic for a certain object.

Feature representation

Methods for representing features suitable for machine learning systems.

Gensim

Python library for text mining purposes.

Graphical user interface

Interface for humans to intact with an electronic device. In contrast to text-based interfaces the graphical user interface contains visual items for interaction.

GUI

see Graphical user interface

Human language technologies

Techniques to automatically process documents containing human language.

Hypernym

Generalisation of a word. e. g. tree $\rightarrow$ forest, woman $\rightarrow$ human.

Hyponym

Subgroup of a word, e. g. tree $\rightarrow$ branch, forest $\rightarrow$ tree.

Information retrieval

The process of information extraction from different resources to satisfy an information need.

Information system

System that tries to cover the need for information by retrieving, producing and computing data and information.

Language model

Statistical model to estimate the underlying word distribution of a text document.

Lemmatization

A method to reduce words to their base form using a vocabulary and a morphological analysis of the words.

LDA

see Latent Dirichlet allocation

Machine learning

Techniques to enable computers to learn without following a deterministic program.

Natural language

The language used by humans.

N-gram

A token consisting of two or more consecutive words from a text.

Perplexity

The perplexity measures how well a probability distribution or probability model predicts a sample.

Part of speech

Words that are grammatically similar.

POS

see Part of speech

Preprocessing

A method to reduce the dimensionality of a text by removing stopwords, lemmatising or stemming and other techniques.

Stemming

A method to reduce words to a word stem by removing suffixes considering certain rules.

Stopword

Stopwords are words that are common in a language and carry little meaning.

Synset

In wordnet a synset is a set of cognitive synonyms of a word.

Text categorisation

Methods to detect similarities within texts or between multiple texts.

Text graph

Representation of a text as a graphs. Words are represented as nodes and the relationships between words are represented as edges.

Text mining

The process of information extraction out of text documents.

TextRank

Algorithm by Mihalcea and Tarau that converts a text into a text graph and then either computes the most relevant keywords or a text summary. [MT04]

Text representation

Feature representation techniques for texts.

Text summarisation

Methods to reduce a text to its most important parts.

Term frequency and inverse document frequency

Method to compute how important a concept is for a text document.

tf-idf

see Term frequency and inverse document frequency

Tokenisation

Methods to split a text into concepts (tokens). These could consist of one or more words.

Topic model

Methods to extract the underlying topics of a text.

Unigram language model

Simplest form of a language model, working with tokens consisting of individual words.

Vector space model

Model to represent text documents as vectors.

VSM

see Vector space model

A. Appendix

A.1. Exemplary Results

A.1.1. Preprocessing

Tokenised text with words to lower case:
'president', 'reagan', "'s", 'approval', 'rating', 'fell', 'after', 'the', 'tower', 'commission', 'criticised', 'his', 'handling', 'of', 'the', 'iran', 'arms', 'scandal', 'a', 'new', 'york', 'times/cbs', 'poll', 'indicates', 'the', 'poll', 'found', 'pct', 'of', 'those', 'surveyed', 'thought', 'he', 'was', 'lying', 'when', 'he', 'said', 'he', 'did', 'not', 'remember', 'if', 'he', 'had', 'approved', 'the', 'original', 'arms', 'sales', 'to', 'iran', 'and', 'pct', 'thought', 'he', 'was', 'telling', 'the', 'truth', 'the', 'poll', 'found', 'pct', 'of', 'those', 'surveyed', 'approved', 'reagan', "'s", 'handling', 'of', 'his', 'job', 'and', 'pct', 'disapproved', 'the', 'approval', 'rating', 'was', 'the', 'lowest', 'since', 'january', 'when', 'pct', 'approved', 'of', 'the', 'way', 'reagan', 'was', 'doing', 'his', 'job'
tokens in text: 96

Tokens and their POS tags after stopword removal:
('president', 'n'), ('reagan', 'n'), ("'s", ''), ('approval', 'n'), ('rating', 'n'), ('fell', 'v'), ('tower', 'a'), ('commission', 'n'), ('criticised', 'v'), ('handling', 'n'), ('iran', 'n'), ('arms', 'n'), ('scandal', 'n'), ('new', 'a'), ('york', 'n'), ('times/cbs', 'n'), ('poll', 'n'), ('indicates', 'v'), ('poll', 'n'), ('found', 'v'), ('pct', 'n'), ('surveyed', 'v'), ('thought', 'v'), ('lying', 'v'), ('said', 'v'), ('remember', 'v'), ('approved', 'v'), ('original', 'a'), ('arms', 'n'), ('sales', 'n'), ('iran', 'v'), ('pct', 'v'), ('thought', 'n'), ('telling', 'v'), ('truth', 'n'), ('poll', 'n'), ('found', 'v'), ('pct', 'n'), ('surveyed', 'v'), ('approved', 'v'), ('reagan', ''), ("'s", 'v'), ('handling', 'n'), ('job', 'n'), ('pct', 'n'), ('disapproved', 'v'), ('approval', 'n'), ('rating', 'n'), ('lowest', 'a'), ('since', 'r'), ('january', 'n'), ('pct', 'n'), ('approved', 'v'), ('way', 'n'), ('reagan', 'n'), ('job', 'n')
remaining tokens: 56

Tokens after lemmatization:
'president', 'reagan', 'approval', 'rating', 'fell', 'tower', 'commission', 'criticise', 'handling', 'iran', 'arm', 'scandal', 'new', 'york', 'times/cbs', 'poll', 'indicate', 'poll', 'find', 'pct', 'survey', 'think', 'lie', 'say', 'remember', 'approve', 'original', 'arm', 'sale', 'iran', 'pct', 'thought', 'tell', 'truth', 'poll', 'find', 'pct', 'survey', 'approve', 'handling', 'job', 'pct', 'disapprove', 'approval', 'rating', 'low', 'since', 'january', 'pct', 'approve', 'way', 'reagan', 'job'
number of tokens in output: 53

A.1.2. tf-idf

Tokens after preprocessing text one: 53
Tokens after preprocessing text two: 53

33% of unique tokens: 12
33% of unique tokens: 15

tf-idf values for text one:
('president', 0.09901475429766744), ('reagan', 0.19802950859533489),
('approval', 0.19802950859533489), ('rating', 0.19802950859533489),
('fell', 0.09901475429766744), ('tower', 0.09901475429766744),
('commission', 0.09901475429766744), ('criticise', 0.09901475429766744),
('handling', 0.19802950859533489), ('iran', 0.19802950859533489),
('arm', 0.19802950859533489), ('scandal', 0.09901475429766744),
('new', 0.09901475429766744), ('york', 0.09901475429766744),
('times/cbs', 0.09901475429766744), ('poll', 0.2970442628930023),
('indicate', 0.09901475429766744), ('find', 0.19802950859533489),
('pct', 0.4950737714883372), ('survey', 0.19802950859533489),
('think', 0.09901475429766744), ('lie', 0.09901475429766744),
('remember', 0.09901475429766744), ('approve', 0.2970442628930023),
('original', 0.09901475429766744), ('sale', 0.09901475429766744),
('thought', 0.09901475429766744), ('tell', 0.09901475429766744),
('truth', 0.09901475429766744), ('job', 0.19802950859533489),
('disapprove', 0.09901475429766744), ('low', 0.09901475429766744),
('since', 0.09901475429766744), ('january', 0.09901475429766744),
('way', 0.09901475429766744)

tf-idf values for text two:
('egypt', 0.12216944435630522), ('allow', 0.12216944435630522),
('five', 0.24433888871261045), ('fugitive', 0.12216944435630522),
('libyan', 0.36650833306891567), ('soldier', 0.12216944435630522),
('land', 0.12216944435630522), ('military', 0.12216944435630522),
('plane', 0.24433888871261045), ('far', 0.12216944435630522),
('south', 0.12216944435630522), ('country', 0.12216944435630522),
('last', 0.12216944435630522), ('night', 0.12216944435630522),
('stay', 0.12216944435630522), ('fly', 0.24433888871261045),
('cairo', 0.12216944435630522), ('official', 0.24433888871261045),
('source', 0.24433888871261045), ('appear', 0.12216944435630522),
('government', 0.12216944435630522), ('agree', 0.12216944435630522),
('demand', 0.12216944435630522), ('two', 0.12216944435630522),
('officer', 0.12216944435630522), ('three', 0.12216944435630522),
('private', 0.12216944435630522), ('political', 0.12216944435630522),
('asylum', 0.12216944435630522), ('immediate', 0.12216944435630522),
('announcement', 0.12216944435630522), ('egyptian', 0.12216944435630522),
('serviceman', 0.12216944435630522), ('north', 0.12216944435630522),
('abu', 0.12216944435630522), ('simbel', 0.12216944435630522),

('c-130', 0.12216944435630522), ('transport', 0.12216944435630522), ('status', 0.12216944435630522), ('sixth', 0.12216944435630522), ('board', 0.12216944435630522), ('pilot', 0.12216944435630522), ('immediately', 0.12216944435630522), ('know', 0.12216944435630522)

Tokens with the 33% highest tf-idf value for text one:
'pct', 'poll', 'approve', 'reagan', 'approval', 'rating', 'handling', 'iran', 'arm', 'find', 'survey', 'job'
Tokens with the 33% highest tf-idf! (tf-idf!) value for text two:
'libyan', 'five', 'plane', 'fly', 'official', 'source', 'egypt', 'allow', 'fugitive', 'soldier', 'land', 'military', 'far', 'south', 'country'

A.1.3. TextRank

Tokens after preprocessing text one: 53
Tokens after preprocessing text two: 53

33% of best nodes: 11
33% of best nodes: 13

33% best keywords for text one:
'approve', 'reagan', 'approval', 'rating', 'pct', 'survey', 'poll', 'handling', 'iran', 'arm', 'job'
33% best keywords for text two:
'approve', 'reagan', 'approval', 'rating', 'pct', 'survey', 'poll', 'handling', 'iran', 'arm', 'job'

A.1.4. LDA

Topics for whole corpus with probability for each word:
topic1: (0, '0.013*"libyan" + 0.013*"five" + 0.013*"official" + 0.013*"plane" + 0.013*"fly" + 0.013*"say" + 0.013*"source" + 0.013*"far" + 0.013*"pilot" + 0.013*"simbel"')
topic2: (1, '0.013*"poll" + 0.013*"pct" + 0.013*"approve" + 0.013*"find" + 0.013*"iran" + 0.013*"rating" + 0.013*"reagan" + 0.013*"survey" + 0.013*"arm" + 0.013*"say"')
topic3: (2, '0.075*"pct" + 0.046*"approve" + 0.046*"poll" + 0.032*"approval" + 0.032*"job" + 0.032*"handling" + 0.032*"arm" + 0.032*"survey" + 0.032*"reagan" + 0.032*"rating"')
topic4: (3, '0.013*"libyan" + 0.013*"five" + 0.013*"source" + 0.013*"plane" + 0.013*"official" + 0.013*"say" + 0.013*"fly" + 0.013*"three" + 0.013*"sixth" + 0.013*"immediate"')
topic5: (4, '0.046*"libyan" + 0.032*"say" + 0.032*"source" + 0.032*"fly" + 0.032*"plane" + 0.032*"official" + 0.032*"five" + 0.017*"serviceman" + 0.017*"military" + 0.017*"officer"')

topics for text one:
topic1: 'pct', 'poll', 'approve', 'iran', 'survey', 'rating', 'arm', 'approval', 'reagan', 'find'
topics for text two:
topic1: 'libyan', 'official', 'five', 'source', 'fly', 'plane', 'say', 'last', 'stay', 'three'

A.1.5. Resulting Differences

f-idf, simple differentiation: 'pct', 'poll', 'approve', 'reagan', 'approval', 'rating', 'handling', 'iran', 'arm', 'find', 'survey', 'job'

tf-idf, semantic differentiation: 'pct', 'poll', 'approve', 'reagan', 'approval', 'rating', 'handling', 'iran', 'arm', 'find', 'survey'

TextRank, simple differentiation: 'approve', 'reagan', 'approval', 'rating', 'pct', 'survey', 'poll', 'handling', 'iran', 'arm', 'job'

TextRank, semantic differentiation: 'approve', 'reagan', 'approval', 'rating', 'pct', 'survey', 'poll', 'handling', 'iran', 'arm', 'job'

LDA with five generated topics before difference extraction:
topic1: 'pct', 'poll', 'approve', 'handle', 'survey', 'job', 'find', 'reagan', 'approval', 'arm'
LDA with five generated topics after difference extraction:
topic1: 'pct', 'poll', 'approve', 'handle', 'survey', 'job', 'find', 'reagan', 'approval', 'arm'

LDA with two generated topics before difference extraction:
topic1: 'pct', 'libyan', 'say', 'approve', 'poll', 'source', 'plane', 'five', 'fly', 'official'
topic2: 'pct', 'poll', 'approve', 'iran', 'survey', 'job', 'arm', 'handle', 'reagan', 'say'
LDA with two generated topics after difference extraction:
topic1:

LDA with nine generated topics before difference extraction:
topic1: 'pct', 'approve', 'poll', 'job', 'find', 'survey', 'approval', 'arm', 'reagan', 'rating'
LDA with nine generated topics after difference extraction:
topic1: 'pct', 'approve', 'poll', 'job', 'find', 'survey', 'approval', 'arm', 'reagan', 'rating'

A.2. Listings

```
# from heapq import nlargest
# from operator import itemgetter
# from collections import Counter
# from gensim import corpora
# from gensim import models

# compute number_of_best_words for both texts and convert to integer, only
    whole words can be retrieved
number_of_best_words1 =
    int(round(len(Counter(text_one))*ratio_of_best_words))
number_of_best_words2 =
    int(round(len(Counter(text_two))*ratio_of_best_words))

# merge both texts to one corpus
corpus = [text_one, text_two]

# map every unique word to a id
dictionary = corpora.Dictionary(corpus)

# convert the corpus to a bag of words
corpus = [dictionary.doc2bow(text) for text in corpus]

# train the transformation model on the corpus, from now on tfidf is a
    read-only object
tfidf = models.TfidfModel(corpus)

# apply tf-idf to whole corpus
corpus_tfidf = tfidf[corpus]

# split corpus back in documents
tfidf_one = corpus_tfidf[0]
tfidf_two = corpus_tfidf[1]

# make dictionary to an actual dictionary and switch key and value
dictionary = dictionary.token2id
dictionary = {y: x for x, y in dictionary.items()}

# map ids back to words
tfidf_one = [(dictionary[word], tfidf) for (word, tfidf) in tfidf_one]
tfidf_two = [(dictionary[word], tfidf) for (word, tfidf) in tfidf_two]

# get the defined number word with the biggest tf-idf
# the tf-idf value is on position 1
one_largest = nlargest(number_of_best_words, tfidf_one, key=itemgetter(1))
two_largest = nlargest(number_of_best_words, tfidf_two, key=itemgetter(1))

# drop the tf-idf value
one_largest = [word for (word, tfidf) in one_largest]
two_largest = [word for (word, tfidf) in two_largest]
```

Listing 1: tf-idf concept extraction

```
# from Approaches import Util

# simple differentiation
if differntiation_method == 'simple':
   return util.simple differntiation(one_largest, two_largest)

# semantic differentiation
elif differntiation_method == 'semantic':
   return util.semantic_differentiation(one_largest, two_largest)
```

Listing 2: tf-idf approaches

```
# from Approaches import Util

# preprocess texts and join list back to string
text_one = '␣'.join(Util.preprocess(path_one))
text_two = '␣'.join(Util.preprocess(path_two))
```

Listing 3: TextRank preprocessing

```
# import gensim

# extract keywords from text, the number of keywords depends on the ratio of the text length
keyword_one = gensim.summarization.keywords(text_one, ratio=ratio_of_text).split()
keyword_two = gensim.summarization.keywords(text_two, ratio=ratio_of_text).split()
```

Listing 4: TextRank keyword extraction

```
# from Approaches import Util

# simple differntiation
if differntiation_method == 'simple':
   return util.simple differntiation(keyword_one, keyword_two)

# semantic differntiation
elif differntiation_method == 'semantic':
   return util.semantic_differentiation(keyword_one, keyword_two)
```

Listing 5: TextRank approaches

```
# import gensim

# merge both texts to one corpus
corpus = [text_one, text_two]

# map every unique word to a id
id2word = gensim.corpora.Dictionary(corpus)

# count the occurrence of every unique word and map the occurrence to the word id (document term matrix)
corpus = [id2word.doc2bow(text) for text in corpus]

# train lda model with whole corpus
lda = gensim.models.ldamodel.LdaModel(corpus=corpus, id2word=id2word, num_topics=number_of_topics, passes=10)

# turn both texts to bag of words
text_one = id2word.doc2bow(text_one)
text_two = id2word.doc2bow(text_two)

# compute the topics for the documents
lda_one = lda[text_one]
lda_two = lda[text_two]
```

Listing 6: LDA topic extraction

```
# import re
# import gensim

same = False
single_words = []
result = []

# convert the number tuple to the words of the topics
topics_one = [lda.print_topic(topic[0]) for topic in lda_one]
topics_two = [lda.print_topic(topic[0]) for topic in lda_two]

# remove everything that do not start with a letter and tokenize, for
    better readability
topics_one = [re.findall('[a-z]+[-,/]*', topic) for topic in topics_one]
topics_two = [re.findall('[a-z]+[-,/]*', topic) for topic in topics_two]

# preprocess again LDA gets  not preprocessed tokens while computing
    topics for the documents
topics_one = [Util.preprocessing('␣'.join(topic)) for topic in topics_one]
topics_two = [Util.preprocessing('␣'.join(topic)) for topic in topics_two]

# remove all words from the topics in topics_one that are in topics_one
    and topics_two
for topic in topics_one:
   for word in topic:
      for other_topic in topics_two:
         for other_word in other_topic:
            if word == other_word:
               same = True
      if not same:
         single_words.append(word)
      same = False
   if len(single_words) > 0:
      result.append(single_words)
   single_words = []

return result
```

Listing 7: LDA difference extraction

```python
# from nltk.stem import WordNetLemmatizer
# from nltk.corpus import stopwords

# convert to lowercase, tokenize text, remove everything that's not a word
text = text.lower()
text = nltk.word_tokenize(text)
text = [word for word in text if any(letter.isalpha() for letter in word)]

# tag words with part of speech tags
text = nltk.pos_tag(text)

# change tags to wordnet tags
text = [(word, get_wordnet_pos(pos)) for (word, pos) in text]

# remove stopwords
wo_stop = [(word, pos) for (word, pos) in text if word not in
    stopwords.words('english')]

# remove all untagged words
wo_stop = [(word, pos) for (word, pos) in wo_stop if pos != '']

# remove every token < 3 characters
wo_stop = [(word, pos) for (word, pos) in wo_stop if len(word) >= 3]

# lemmatize the text with word net
wordnet_lemmatizer = WordNetLemmatizer()
lemma = [wordnet_lemmatizer.lemmatize(word, pos=pos) for (word, pos) in
    wo_stop]

return lemma
```

Listing 8: Preprocessing

```python
# from nltk.corpus import wordnet

noun = ['CD', 'FW', 'NN', 'NNS', 'NNP', 'NNPS']
adjective = ['JJ', 'JJR', 'JJS']
adverb = ['PDT', 'IN', 'DT', 'MD', 'RB', 'RBR', 'RBS', 'RP']
verb = ['VB', 'VBD', 'VBG', 'VBN', 'VBP', 'VBZ']

if pos_tag in noun:
   return 'n'
elif pos_tag in adjective:
   return 'a'
elif pos_tag in adverb:
   return 'r'
elif pos_tag in verb:
   return 'v'
else:
   return ''
```

Listing 9: Treebank to wordnet POS

```python
# from nltk.corpus import wordnet

word_synonyms = []

# find all synonyms for word one
for synset in wordnet.synsets(word_one):

   # don't write the word itself in the list
   for lemma in synset.lemma_names():
      if lemma != word_one:
         word_synonyms.append(lemma)

# check if second word is in the synonym list of the first one
if word_two in word_synonyms:
   return True
else:
   return False
```

Listing 10: Finding synonyms

```python
# from nltk.corpus import wordnet

word_hypernyms = []

if len(wordnet.synsets(word_one)) > 0:

   # find all hypernyms of the first word
   for words in wordnet.synsets(word_one)[0].hypernyms():

      # don't write the word itself in the list
      for lemma in words.lemma_names():
         if lemma != word_one:
            word_hypernyms.append(lemma)

      # check if second word is in the hypernym list of the first one
   if word_two in word_hypernyms:
      return True
   else:
      return False
else:
   return False
```

Listing 11: Finding hypernyms

```python
# from nltk.corpus import wordnet

if len(wordnet.synsets(word_one)) > 0:

   # find all hyponyms of the first word
   for words in wordnet.synsets(word_one)[0].hyponyms():

      # don't write the word itself in the list
      for lemma in words.lemma_names():
         if lemma != word_one:
            word_hyponyms.append(lemma)

   # check if second word is in the hyponym list of the first one
   if word_two in word_hyponyms:
      return True
   else:
      return False
else:
   return False
```

Listing 12: Finding hyponyms

```python
# remove words that are in both lists
result = [word for word in list_one if word not in list_two]

# return the differences
return two
```

Listing 13: Simple differentiation: filtering same words

```python
# from nltk.corpus import wordnet

result = []
same = False

# remove all words that are synonyms, hypernyms or hyponyms to one another
for word in list_one:
   for word2 in list_two:
      if check_wordnet_synonym(word, word2) or
          check_wordnet_hypernym(word, word2) or
          check_wordnet_hyponym(word, word2):
         same = True
   if not same:
      result.append(word)
   same = False

# return the differences
return result
```

Listing 14: Semantic differentiation: filtering synoynms, hypernyms and hyponyms

A.3. LDA Topic Output

Text one:
AMYLUM CHAIRMAN DISAPPOINTED BY FERRUZZI-CPC DEAL
Belgian starch manufacturer <Amylum NV> is surprised and disappointed that its 675 mln dlr offer for the European business of CPC International Inc <CPC.N> was apparently rejected in favour of a lower 630 mln dlr bid by Italy's <Gruppo Ferruzzi>, chairman Pierre Callebaut said. Callebaut told Reuters that Amylum, a leading starch and isoglucose manufacturer in which Britain's Tate and Lyle Plc <TATL.L> holds a 33.3 pct stake, had made an undisclosed initial takeover offer for CPC's European corn wet milling business by the close of CPC's tender on March 17. The offer was raised on March 24 to a final 675 mln dlrs in cash after CPC told Amylum its initial bid was below Ferruzzi's 630 mln stg offer, Callebaut said. On the same day, CPC announced it had agreed in principle to sell its European business to Ferruzzi in a 630 mln dlr deal. Noting that Ferruzzi was studying a public offering of shares in its unit <European Sugar (France)> to fund the CPC takeover, Callebaut said Amylum may still succeed in its bid. "For the time being we just await developments. But I note that whereas our higher offer was in cash, Ferruzzi apparently is still organising finance," Callebaut said.

Text two :
OILMEAL DEMAND STILL STRONG IN U.S., SOVIET UNION
Oilmeal demand remained strong in the United States in July and August and six pct up on levels seen in the same months last year, with most of the rise coming in soymeal, the Hamburg based publication Oil World said. Total U.S domestic usage of the nine major oilmeals rose to a record 19 mln tonnes in October 1986/August 1987, up 4.2 pct on the same year-ago period, with the increase in soymeal at seven pct, it said. Soviet soymeal demand rose by 310,000 tonnes in July and 330,000 tonnes in August over the respective year-ago months following a huge increase in soymeal imports. Oil World presumed some imports were not used immediately but went into stocks. With imports again large in September, it estimated Soviet soymeal stocks sharply up at 800,000 tonnes by October 1 from around 130,000 at the same time last year. EC oilmeal demand rose 100,000 tonnes in August from a year earlier, with soymeal up 45,000 tonnes. Crushings of rapeseed, sunseed and soybeans will probably rise from last year due to bumper EC crops. It estimated the EC rapeseed crop at a record 5.9 mln tonnes, up from 3.7 mln last year. Rapeseed disposals were reported at 2.1 mln tonnes by September 20 against 1.4 mln at that time last year.

Below the the LDA was applied on the texts above with one to ten topics. No mater how many topics are generated only one topic is assigned to the first text. Except if only one topic is generated. The probability of this topic is too low to be assigned to a document.
As the LDA is a probabilistic approach the computation is run for five times. But the results do not differ much between these runs.

```
Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business

Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business

Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business

Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business
Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business
```

```
Number of topics 1: []
Number of topics 2: cpc, callebaut, amylum, offer, say, european, ferruzzi, bid, dlr
Number of topics 3: cpc, amylum, callebaut, offer, say, ferruzzi, european, business, dlr
Number of topics 4: cpc, offer, amylum, callebaut, say, ferruzzi, european, business, dlr
Number of topics 5: cpc, amylum, offer, callebaut, say, european, ferruzzi, business, dlr
Number of topics 6: cpc, offer, amylum, callebaut, ferruzzi, say, european, bid, dlr
Number of topics 7: cpc, amylum, offer, callebaut, ferruzzi, say, european, dlr, bid
Number of topics 8: cpc, amylum, offer, callebaut, ferruzzi, say, european, bid, business
Number of topics 9: cpc, callebaut, amylum, offer, european, say, ferruzzi, bid, business
Number of topics 10: cpc, callebaut, offer, amylum, say, ferruzzi, european, dlr, business
```

A.4. Remarks In Human Evaluation

Difficulties: long scrolling makes concentrating difficult
Remarks: approaches seem to work best if texts have the same length and are relatively short

Difficulties: Texts are long and difficult to read. Lists of words are sometimes long.
Remarks:

Difficulties:
Remarks:

Difficulties: It's difficult to rank the keywords when the words or topics of both texts are very similar and therefore their semantics. The lists of keywords often contains expletives that aren't meaningful.
Remarks:

Difficulties: Texts were sometimes difficult to read because of formatting mistakes or unknown abbreviations. Some groups of keywords contained the same keywords. There was no difference. Some keywords were not meaningful (e.g. may, also).
Remarks:

Difficulties: Some of the approaches had either an equal amount of words or an indentical wordlist, so I couldn t rate them from 1-5.
Remarks:

Difficulties: Some word-listings were way to long to differentiate them from others.
Remarks: I dont see the sense in compairing one sentence to a whole text page, the topics could have been more interesting.

Difficulties: The evaluation used mostly trading-related newspaper articles, which are sometimes difficult to parse for a non-native reader, both because they assume knowledge about trade terms and processes and because they are written in a very compressed style.
Remarks: The graphical interface used a ranking between 1 and 5 (with the information that the same ranking could be applied to multiple results if they were equally good matches). This may lead to "holes" in the ranking (e.g. 1, 3, 4, 4, 5) depending on the user's order of rating the results. To counteract this, future similar evaluations could instead use a drag-and-drop interface to allow ranking the results without explicitly specifying numeric values.

Difficulties: Text should have less numbers and more descriptive content. Most of the texts simply described that some numbers rose and others fell. Not only makes this reading boring more importantly this makes huge parts of the text totally irrelavant.
Remarks: Words that occur near other words should be grouped together in the word list. Without context the lists do often mis-describe the content.

Difficulties: - two identical texts
- a very short text(with only about 5 relevant words) and a long text, which were hardly comparable
- sometimes unclear which aspects to base the ranking/grading on (order of words based on importance / excluded words / included words)

Remarks: - compound words were separated (e.g. indian|summer has a completely different meaning when separated)