CHALMERS

Automatic extractive single document summarization

An unsupervised approach

Master of Science Thesis

Computer Science: Algorithms, Languages and Logic

JONATAN BENGTSSON
CHRISTOFFER SKEPPSTEDT

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
Göteborg, Sweden, May 2012

Automatic extractive single document summarization
An unsupervised approach

J. Bengtsson
C. Skeppstedt

Examiner: Devdatt Dubhashi

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden May 2012

Abstract

This thesis describes the implementation and evaluation of a system for automatic, extractive single document summarization. Three different unsupervised algorithms for sentence relevance ranking are evaluated to form the basis of this system. The first is the well established, graph based TextRank, the second is based on K-means clustering and the third on one-class support vector machines (SVM). Further more, several different variations of the original approaches are evaluated. These algorithms are, in themselves, language independent, but language dependent text preprocessing is needed to use them in this setting. Evaluations of the system, using the de facto standard ROUGE evaluation toolkit, shows that TextRank obtains the best score. The K-means approach gives competitive results, beating the predefined baselines on the main test corpus. The one-class SVM yields the worst performance of the three, but still manage to beat one of two baselines. The system is evaluated for both English and Swedish, however, the main evaluation is done for short news articles in English. In our opinion this system, together with domain specific boosting provides adequate results for the corpora tested.

Generated abstract

What follows is a extractive summary, generated using the system described in this thesis on sentences from the motivation and scope sections.

> A good way to get an overview of a text is to read a summary of it, but writing summaries is a time consuming task. For instance, if one would have a system to generate summaries automatically, those summaries could be used to populate a meta data field in an indexing service. When someone searches that index, the summary could be presented and he/she could determine if a specific document seems interesting. There are many different aspects to the field of document summarization, this thesis will in no means investigate them all. In single document summarization, only one document is used as basis for the summary. Extractive automatic document summarization can be seen as a machine learning task, where a system is to learn what kind of sentences it should extract from a document to form a good summary, given a set of features. An unsupervised approach would require no training set, or knowledge of what sentences "should" be extracted from each document. In conclusion, this thesis addresses the construction and evaluation of a system for automatic, extractive summary generation of single documents. This system is based on three different unsupervised algorithms: TextRank, K-means clustering and one-class SVM.

Acknowledgements

This thesis would not have been what it is without the help and encouragement from the following people.

Thank you Svetoslav Marinov, our supervisor at Findwise, for your willingness to share your expertise, providing us with resources and ideas, proof reading this thesis and most of all for your positive encouragement.

Thank you Chiranjib Bhattacharyya, our supervisor at Chalmers, for your interesting ideas and suggestions, for your patience and most of all for your enthusiasm and interest in our work.

Thank you Martin Johansson at Findwise, for introducing us to your work on keyword extraction. Your extraction system forms as the basis for some interesting approaches in this project, which would not have been possible without it.

Thank you Rada Mihalcea for taking the time to answer our questions about details of the TextRank algorithm. More over, thank you for your previous work in the field, it has helped us greatly.

Contents

Terminology

- NLP - Natural Language Processing, a field of Computer Science, Artificial Intelligence and Linguistics, concerned with the understanding/interpretation of human language by computers.
- Document - For the purpose of this report, a document is a textual entity consisting of a title and a collection of sentences (for example a news article).
- Corpus - A collection of documents (plural corpora).
- PoS-tag - Part of Speech tag, a tag marking the lexical class of a word, such as nouns, verbs, adjectives, etc.
- Clustering - The task of grouping similar entities into coherent groups, based on the features of the entities.
- Tokenization - The task of dividing a text string into meaningful entities/tokens, such as words, numbers, punctuations etc.
- NP-chunking - Noun Phrase chunking, the task of identifying noun groups (phrases) in a text string.
- Noun phrase - A phrase built around a noun, consisting of a noun/pronoun and words that "modify" that noun/pronoun.
- NER - Named Entity Recognition, the task of extracting all names (persons, places, organizations etc.) form a text string.
- Dependency structure - A linguistic analysis of a sentence, represented as a graph, where nodes are the tokens and edges are grammatical relations.
- Stemming - The task identifying the stem or "root" of a word, often by removing its affix.
- Keyword - For the purpose of this report, a word or phrase of high importance to the content of a certain document.
- N-gram - A n-gram is a sequence of n consecutive text entities in a text, in this report the entities are words.

Chapter 1

Introduction

This thesis was written as a master thesis at Chalmers University of Technology and Findwise AB, Göteborg, Sweden in the spring of 2012. The thesis addresses the implementation and evaluation of a system for automatic extraction of summaries from single documents.

1.1 Motivation

In today's society people have access to vast amounts of textual information by the press of a button. This calls for ways to evaluate the relevance of text documents without reading the entire text. A good way to get an overview of a text is to read a summary of it, but writing summaries is a time consuming task. One way to address this problem is automatic summary generation.

For instance, if one would have a system to generate summaries automatically, those summaries could be used to populate a meta data field in an indexing service[1]. When someone searches that index, the summary could be presented and he/she could determine if a specific document seems interesting.

Automatic document summarization is an advanced topic, touching several other fields in computer science such as NLP, machine learning and algorithm analysis. Even though deep understanding of these fields is not needed to understand this report, one needs knowledge in these subjects to construct a summarization system.

1.2 Scope

There are many different aspects to the field of document summarization, this thesis will in no means investigate them all. What follows is a definition of the scope and limitation of the study conducted.

There are two major categories of automatically generated summaries, abstractive and extractive summaries. These two concepts will be explained further in chapter 2. Only extractive summarization will be evaluated in this report.

Summarization of texts can be done on different levels. Depending on the application, it might be relevant to take either one or several documents into consideration to create a summary. In multi

[1] Search engine indexing collects, parses, and stores data to facilitate fast and accurate information retrieval[7].

document summarization, the idea is to take several documents describing, for instance, the same event as input and return a summary of that event. In single document summarization, only one document is used as basis for the summary. One of the goals of the work underlying this report is to be able to summarize documents to be used in an indexing service, this means only the subject of single document will be investigated in this report.

Extractive automatic document summarization can be seen as a machine learning task, where a system is to learn what kind of sentences it should extract from a document to form a good summary, given a set of features. A supervised machine learning approach requires the use of a good training set consisting of text documents paired with "gold standard" sentences from each document. An unsupervised approach would require no training set, or knowledge of what sentences "should" be extracted from each document. As such training sets proved hard to come by, this thesis focuses solely on unsupervised approaches.

More specifically, this thesis will address the implementation and evaluation of three unsupervised algorithms. The first one, TextRank, is a well know algorithm that have been shown to performed well in the past[46]. The second, based on k-means clustering, is highly influenced by previous research in the field providing promising results[30]. However, the feature representation forming the basis for the clustering in this approach differs significantly from that of the previous work. The third approach, based on one-class Support Vector Machines (SVM) have, to our best knowledge, not been tested in the setting of automatic single document summarization before. TextRank and one-class SVM will be varied by evaluation of several different similarity measures for sentences, while K-means clustering will be varied by several different feature representations.

The three unsupervised algorithms are by themselves language independent, but dependent on language specific preprocessing. This report will only address evaluation for summary generation in English and Swedish.

In conclusion, this thesis addresses the construction and evaluation of a system for automatic, extractive summary generation of single documents. This system is based on three different unsupervised algorithms: TextRank, K-means clustering and one-class SVM.

1.3 Purpose and goals

The purpose of the current work is to investigate the possibility of constructing a system for extractive single document summarization, with the use of state of the art approaches.

One of the goals is to produce "good enough" summaries to be used as part of a indexing or query service for a collection of text documents. What "good enough" means is, of course, subjective and a bit hard to define. For the purpose of this report, a "good enough" summary consists of sentences which preserves the most important information of the original document. Evaluation of this property will be done by manual exploration of random examples.

A more objective goal is to implement a system, that outperforms certain predefined baselines, using a standard evaluation method as measurement. The toolkit used for evaluation in this report is ROUGE. ROUGE is an evaluation system for comparing generated summaries to "gold standard" summaries[1]. Two baselines were used for evaluation in this report. The first is the baseline of just selecting sentences at random from the text to form a summary of a certain length, the second is to use the start of a text as a summary.

Using the metrics stated above, evaluation of the three unsupervised algorithms aim at answering the following questions:

- Can the results of TextRank be improved by variations of the algorithm? Variations include different similarity comparers (section 4.2) and domain specific boosting (section 4.5.3).

- Can the k-means and SVM based algorithms outperform the well established TextRank algorithm? Can they outperform the baselines?
- Can the three algorithms outperform the baselines for a Swedish corpus?

1.4 Method

Before any work could be initiated on implementing the summary system, a literary survey was conducted to decide on what approaches might be fruitful. Even thought this survey was initiated before starting the implementation phase, research of previous, relevant work has been an ongoing process through out the whole project, as problems and new ideas for improvement emerged.

After the initial research phase, the implementation and evaluation phase began. Each approach, that was to be tested, was implemented in Java and evaluated using the ROUGE evaluation toolkit. To be able to implement the solutions in Java the following additional libraries were used:

- Apache OpenNLP - A Natural Language Processing library, providing functionality for many NLP tasks, such as PoS-tagging, NP-chunking and tokenization[11].
- Stanford CoreNLP - A Natural Language Processing library, providing similar functionality to OpenNLP, used in this project mainly for named entity recognition[10].
- MaltParser - A system for dependency parsing, used in this project to construct dependency structures from sentences[15].
- Apache Commons Math - A extensive math library, used in this project mainly for functionality regarding clustering and linear algebra[3].
- LIBSVM - A multi lingual Support Vector Machine library[25].
- Findwise AB proprietary software - Used in this project for stemming and keyword extraction.

After the different approaches were implemented and extensively evaluated (using the ROUGE toolkit), this report was written.

1.5 Results

The system have been extensively tested and evaluated using the ROUGE evaluation toolkit. Of the three algorithms evaluated, TextRank outperforms the other two for the corpora used for both English and Swedish. The K-means algorithm performed well, providing results better than both baseline algorithms. The one-class SVM performed worst of the three algorithms and only managed to outperform the baseline of selecting sentences at random for the English corpus. When it came to the evaluation for Swedish, all the implemented algorithms provided better results than those of the baselines.

Exploitation of domain knowledge, such as sentence position relevance, improved the evaluation score of the TextRank and K-means algorithms. Such domain specific boosting was not explored for one-class SVM. The various measures of sentences similarity tested for the TextRank and one-class SVM algorithms did not make as big an impact as expected.

The results observed for the English corpus was compared to those presented in two other papers[46, 30]. The results presented in these papers closely related to what was observed for the TextRank and K-means algorithms in this report, with the one-class SVM falling a bit short.

1.6 Outline

The outline of this report will follow the structure of:

- Introduction
- Document summarization
- Methodology
- Implementation
- Experiments
- Comparison and results
- Future work
- Conclusion

The document summarization section describes the field of automatic document summarization and the challenges of evaluating the quality of a summary. The methodology and implementation parts aim to describe the implemented system visualized in fig. 1.1. Specifically, the methodology will give a description of the general algorithms used for sentence extraction, while the implementation part will focus more on customization for the given task of document summarization.

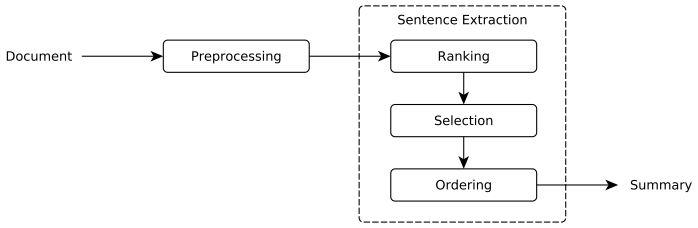


Figure 1.1: System overview

In the experiments section, the evaluation results for the different algorithms will be presented. In the comparison and results section, these results will be compared to each other as well as two predefined baselines and also against the results of two other papers influential to this thesis[46, 30].

The next part of the thesis will address future work. Solutions that were not fully or satisfactory implemented, due to lack of time and/or resources will be discussed here.

The report will be concluded with a section tying the results to the purpose and goals presented in the introduction, to evaluate the success of the project.

Chapter 2

Automatic document summarization

This chapter aims at providing a brief overview of the field of automatic document summarization. The works most influential to this thesis will be presented in some what more detail. It will also address some of the difficulties involved with evaluating the quality of a summary.

2.1 Field overview

The field of automatic summary creation is now more than fifty years old. Early research centered around word frequency analysis of the words in scientific articles to form extractive summaries[43]. Later more features, such as sentence position and words shared with document title, were introduced for the representation of sentences to improve the quality of the sentence extraction[28]. The introduction of multi feature representation of sentences lead the way of the use of more recent, machine learning based approaches in the field[48].

Originally the motivation for research in the field was applications for the digitization of books and scientific papers. With the arrival of the Internet and what it meant for information availability, automatic summarization found new applications and the subject received renewed interest[31]. The redundancy of the information in documents, accessible on the web, spawned the interest of a new direction in the field: multi document summarization[48].

As a result of the growing interest in automatic summarization, a new conference addressing the subject was formed in 2001. The Document Understanding Conference (DUC)[4] hosted annual competitions in tasks concerning automatic summarization and summary evaluation, open for anyone interested between 2001 and 2007. In 2008 DUC became a "summarization track" in the Text Analysis Conference (TAC)[17]. During the first two years of its existence, single document summarization of news articles was one of the tasks addressed in DUC[48]. Other than pushing the research in this specific area, it also resulted in the construction of corpora used not only for the competitions, but also for evaluation purposes in later works[46, 30, 21].

Since its cancellation as a competition track in DUC in 2002, several papers regarding single document summarization have been published. Some recent works aim at a taking advantage of domain specific knowledge, to form supervised summarization systems[55, 51]. Other systems aim at both domain and (near) language independence[46, 30, 31].

The large majority of the work done in the field of automatic summarization this far have only considered the construction of extractive summaries. In extractive summarization passages from the original document (such as sentences or paragraphs) are reused in the generated summary. In

contrast abstractive summarization is more similar to how humans write summaries, meaning new content is introduced to generate a shorter version of the original text. Even though some work have been done concerning the merging and post-editing of sentences, truly abstractive summarization depends on machine understanding not yet reached in the field of Artificial Intelligence.

2.2 Related work

This thesis does in no way aim at presenting an extensive study of the entire field of automatic summarization. The core part of this project is the creation of a unsupervised, extractive, automatic summarization system. The algorithms used in this system heavily relies on results and ideas presented in the following works:

- TextRank (2004)[46]. A graph based algorithm for ranking entities in texts based on Google's PageRank[24]. The algorithm can be used not only for summary creation, but also for other NLP tasks, such as keyword extraction.
- K-means clustering for automatic single document summarization (2008)[30]. Even though k-means clustering is a well known algorithm in unsupervised machine learning, it is, to our knowledge, relatively unexplored when it comes to single document summarization. In 2008, however, some promising results where presented on the subject.

2.3 Evaluation

If we can evaluate how well a summarization system is performing, we can measure how different options improve the produced summaries. This is also essential in order to compare it against other systems. But measuring the quality of a summary is hard and has become its own separate problem in the field of text summarization. Some of the difficulties will be discussed here. Furthermore, in order to put the results of this report in the context of other relevant work, the evaluation method and corpus must be carefully chosen.

2.3.1 Difficulties

If extractive golden summaries are available, the sentence extraction can be seen as the task of classifying a sentence as being in the summary versus not being in the summary. It would be reasonable to use precision and recall, which are common measures of classification accuracy. In this context, precision and recall would be defined as follows:

$$P = \frac{|\text{system sentences} \cap \text{gold standard sentences}|}{|\text{system sentences}|}$$
$$R = \frac{|\text{system sentences} \cap \text{gold standard sentences}|}{|\text{gold standard sentences}|}$$

It is common to combine precision and recall into an harmonic mean known as F_1:

$$F_1 = 2\frac{P * R}{P + R}$$

But how are extractive gold standards produced? Using a reference system is one way, but the more common way is to have humans produce them[31], although it comes with some caveats. For a large corpus, it is both a time consuming and expensive task to solve using human labor.

An even greater problem is that different human summarizers generally select different sentences, and even the same human summarizer might pick different gold standards at different times for the same document. An experiment on the subject where 6 human summarizers were asked to pick the 20 most representative sentences from a document had only 8% overlap on average[49]. In another experiment, 5 summarizers were asked to pick the 20 most representative sentences from six scientific journal articles. Two months later they were asked to repeat the same task, as well as mark which sentences they thought they had picked the first time. The results showed that they recalled on average 42% of their first picks[50].

Clearly, human inconsistency is a big problem in this case. One way to mitigate the problem is to let multiple human summarizers vote on which sentences they think should be included in the summary, and then produce a gold summary based on the votes. An effort for such a corpus called *KTH eXtract Corpus* has been made, but it only contains Swedish texts and is still relatively small[16].

There is another problem with using F_1 score on a sentence level, because it might not be the best level of granularity[48]. Consider the gold standard of 2 sentences: a) *"There was a fire"* and b) *"Several houses were damaged in a fire"*. Then consider two different systems where the first system picks a) and the second system picks b). They will both have the same F_1 score, but b) is arguably a more informative sentence, and the second system has picked a summary that better relates to the gold standard. Strictly *recall*-oriented evaluation has been proposed instead of the F_1 score, but the problems with human disagreement still remains, and does not solve the example given here[27].

Another option for the evaluation is to use abstractive gold standard summaries instead. These kind of corpora are easier to come by, however F_1 score on the sentence level is infeasible for such corpora. Abstracts generally contain sentences that are not found in the document, which makes it difficult to calculate *precision* and *recall* scores on the sentence level directly, thus evaluation must be measured in some other way. There are suggestions on how to measure extractive summaries against abstractive gold standards, but many of them depend on varying amounts of manual labor[48, 54]. Abstractive summaries are of course also subject to human disagreement.

2.3.2 ROUGE

ROUGE is the most commonly used tool to produce metrics of summarization performance[48]. Most of the literature that was studied reported the results of their proposed systems in ROUGE score figures[39, 21, 38, 30, 46, 29, 31] and can be considered a de facto standard to evaluate text summarization systems. It is designed to determine the quality of summaries against gold standards by counting overlapping units (such as word sequences or N-grams)[40, 41], whereas using the 1-grams setting has been shown to agree the most with human judgments[30, 41]. A common notion of the 1-gram setting is Ngram(1,1), which is how it will be referred to in the rest of this thesis. Unlike its predecessor BLEU, which is also based on N-gram co-occurrence, it is *recall*-oriented which is preferred in summarization evaluation[48, 41]. The evaluation is completely based on N-gram coverage, no other sophisticated evaluation is performed, which is a weakness that has been pointed out compared to other evaluation methods[48, 54, 53].

It should be noted that ROUGE is an *intrinsic* evaluation tool, i.e. it only measures the system summaries against golden standard abstracts, and does not consider any target audience[31]. The contrast is *extrinsic* evaluation, which measures the impact a summary has on some task. For

instance, how a summary may assist a user in deciding if a given document is relevant to some topic or not[48].

Some work has been done to see how ROUGE can be tricked to give a very good score, although the quality of the summary is considered bad[53]. The system proposed in [53] exploits the fact that ROUGE counts bigram occurrences, and starts by collecting bigram frequencies from the text. Then it uses a Markov model to select bigrams to include in the summary, thus it is not an extractive summary since it does not pick entire sentences. An excerpt of a summarization made by this system is presented here:

> of the hurricane andrew had been injured and the storm caused by the gulf of mexico and louisiana and at least 150 000 people died on the us insurers expect to the florida...

While it gives some idea of what the text is about, it can hardly be considered a good summary from a human point of view. It acquires a very high *recall*-score, and is especially successful in the earlier versions of ROUGE where only *recall* was considered. However, the results in this paper are based on a newer version that calculates F_1 scores. Although this exploit achieves a very high *recall* score, it also has a very low *precision* score, which results in a low F_1 in total.

ROUGE is not perfect, but it seems to have been adopted as "good enough" and de facto standard to give an accessible measure on the performance of a summarization system. The details and theory of ROUGE will not be further discussed here, but an extensive description on how it was utilized is continued in appendix A.

2.3.3 Corpora

To be able to evaluate system performance and compare the system to competitors, the choice of corpus is important. The documents of the corpus should be easy to extract and parse into sentences. The number of documents in the corpus is also of some significance, since the summarization system should be evaluated on average on many documents. Furthermore, to be able to benchmark the system against competitors, the corpus has to be used in relevant work in the field.

English corpus

From the literature study it became apparent that for single document summarization, the corpus of the 2002 Document Understanding Conference was still commonly used[30, 21, 46], especially in combination with the ROUGE evaluation system. The DUC2002 corpus contains 567 documents that comes in two versions; a simple version where only title, meta data and text content is marked up (using an XML namespace called "TREC tags"[13]), and a preprocessed version where also the sentences have been split and marked up. The sentence splitting appears to be machine made, since obvious errors have been found through manual inspection. However, these errors are left as is in order to compare the evaluation results to relevant competitors.

The contents of the corpus are short news articles divided into 59 sets, where each set describes some specific event. The set structure is not exploited by the summarization system in this report, but is more suitable for multi document summarization. Using the preprocessed sentence splitting of the DUC2002 corpus, the number of sentences per document is in the range of 5 to 177, while the average is 29 sentences. The number of words per sentence range between 1 and 286 words, while the average is 19.7 words. By investigation, only 5 sentences exceeds the length of 100 words and turns out to be mistakes made by the sentence splitter. The short sentences seem to be more correct, while some of them also appear to be mistakes made by the sentence splitter, e.g. instead

of considering *"Dr. D. Dignan, who performed surgery..."* as one sentence, it has been split into two sentences: *"Dr."*, and *"D. Dignan, who performed surgery..."*.

Each document of the DUC2002 corpus is complemented with abstracts written by human annotators. Most documents had two abstracts written by different annotators, but 24 documents had only one abstract. The abstracts consist of approximately 100 words according to the corpus description, but range between 80 and 158 words with an average of 113.6 words. The number of sentences in the abstracts range between 3 and 14 sentences with an average of 5.6 sentences.

Swedish corpus

The DUC2002 corpus is exclusively in English, so another corpus had to be found to evaluate the system for Swedish. No Swedish corpus evaluated using ROUGE was found, so there were no competitive results to take into consideration when deciding on the corpus. The idea of using Wikipedia articles emerged, and seemed interesting given that it is a well known corpus, and also very different from DUC2002. Furthermore, featured articles on Wikipedia are well structured and follow certain style guidelines[8], specifically:

> (a) a lead: a concise lead section that summarizes the topic and prepares the reader for the detail in the subsequent sections

This lead section is similar to an abstractive summary of the article, which enables the system to evaluate the Wikipedia corpus in the same way as DUC2002. There is a Perl script for the purpose of scraping featured Wikipedia articles[12]. Unfortunately, the markup of Wikipedia seems to have changed since the script was written, which makes it fail. A previously scraped corpus from 2010 containing 251 articles in Swedish was acquired from the author of the script. A scraped article is an XML file where the summary (lead section) and content is separated by markup. Since the individual sentences are not marked up, sentence splitting is applied to provide input data to the preprocessing steps.

The article lengths range from 62 to 564 sentences with an average of 200 sentences. The sentence lengths range from 1 to 317 words with an average of 21 words (comparable with the avg. sentence length of DUC2002). The 317 word sentence was investigated and found to be a long, semicolon separated enumeration. If it is disregarded, the longest sentence is 126 words. In contrast to news articles, the Wikipedia articles are expected to contain longer and more complex sentences. The abstracts are not constrained to approximately 100 words, as in DUC2002, but range from 33 to 638 words, with an average of 217 words. In terms of of sentences, the abstracts range from 1 to 34 sentences with an average of 11 sentences.

A summary generated by the system should have approximately the same length as its golden standard. This is the way that the DUC2002 corpus was meant to be used for single document summarization, and also seems like a fair constraint. The length of the abstracts in DUC2002 are on average 113 words with a standard deviation of only 6 words, making it very consistent. The desired length of a system summary can then be set to 100 words. Abstracts in the Wikipedia corpus, however, are on average 217 words with a standard deviation of 120 words. For a document in this corpus, the desired length of its system summary is set to the length of its golden standard abstract.

Chapter 3

Method

This chapter will provide an overview of the algorithms used for sentence extraction in the implemented system. Even though sentence selection and ordering will be addressed, the emphasis of this chapter will describe the three unsupervised algorithms used for sentence ranking: TextRank, K-means clustering and one-class SVM.

3.1 Sentence Extraction

Sentence extraction is the most central step in an automatic, extractive summarization system. The task can be described as, given a document, return the sentences from the document that best describe its content, in a given order. This task can be divided into three distinct parts, sentence ranking, sentence selection and sentence ordering.

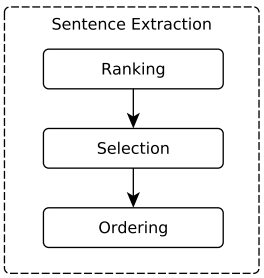


Figure 3.1: The sentence extraction process

3.1.1 Sentence Ranking

Sentence ranking is the process of ranking the sentences in a document in order of importance from a summary point of view. This can be done either by assigning a real number value to each sentence, marking its importance, or by constructing a list from the sentences ordered by importance. Later in this chapter, three different algorithms to perform this step will be explained. As the target application of this system is an indexing service, where summaries for documents are generated offline and then saved in the index, the time complexity of these algorithms will not be the focus of this report.

3.1.2 Sentence Selection

When the sentences have been ranked, the next step in the extraction process is sentence selection. Sentence selection is the task of selecting which sentences should be included in the summary, given a set of ranked sentences from a document.

The main variable to consider when it comes to sentence selection, is the length of the desired summary. For the evaluation of this system, the summary length is measured in words. Another natural way to measure the length of a summary would be number of sentences. The reason number of words is used as the length measurement in this thesis, is to be a able to compare evaluation results with other reports using this measure[30, 46].

Depending on how strictly one views the notion of the desired length of a generated summary, different selection strategies can be applied. If the desired length is seen as a maximum that is not to be exceeded, the task of sentence selection becomes very relevant to the extraction process.

The view of the summary length as a strict maximum gave rise to the use of two different selection strategies in this project. The first one is rather naive and can be described with the following pseudo code:

```
Selection(Set of ordered sentences, max summary length)
    Summary := Empty set of sentences
    for(Sentence S in Set of ordered sentences)
        if(Summary.length + S.length < max summary length)
            Summary.add(S)
        end if
    end for
    return Summary
```

This selection strategy gives priority to highly ranked sentence, but if a highly ranked sentence is too long, the selection method considers shorter, low ranking sentences to "fill up" the summary.

If each sentence in a document is assigned a rank value, one possible goal of the selection step can be to maximize the total ranking value of the sentences selected. This goal can be reached by reducing the sentence selection step to solving the well know knapsack problem. This is the problem of filling a knapsack with objects assigned a value and a weight, in such a way that the total value of the objects in the knapsack is maximized, while the total weight does not exceed a predefined threshold. In the case of sentence selection, the objects to fill the knapsack with are sentences. A sentence objects length is its weight, and the value associated with it is simply its rank value. The problem can be solved in pseudo polynomial time[36]. Even though the algorithm guarantees the optimal total rank value of the solution, it does not guarantee that the highest ranking sentences are included in the summary.

If the desired length of a summary is viewed as a approximation rather than a strict maximum, the sentence selection problem can be solved using the following, more relaxed approach:

```
Selection(Set of ordered sentences, desired summary length)
    Summary := Empty set of sentences
    for(Sentence S in Set of ordered sentences)
        if(Summary.length < desired summary length)
            Summary.add(S)
        else
```

```
            break for
        end if
    end for
    return Summary
```

This approach views the desired summary length as a minimum rather than a maximum, resulting in summaries of the desired length or longer. As this "approximation" view is shared by the papers that are to serve as result comparison for this report[30, 46], this will be the sentence selection strategy used for the evaluation of the system.

3.1.3 Sentence Ordering

When the sentences of the summary have been selected, what remains to decide is in which order they should be presented. The presentation of the extracted sentences have been proven to be important to a readers opinion on the quality of the summary. Experiments have shown that reordering sentences in an extractive summary can change a readers opinion of it from "poor" to "good" (in the scale "poor", "fair", "good")[22]. Sentence ordering is an even greater problem in multi document summarization, where no single ordering from the original text(s) can be used.

In this project, not a lot of effort was put into the task of sentence ordering. The selected sentences are simply presented in the order they appear in the original document. Depending on the application, other ordering strategies such as relation to a search query, or in order of ranking "importance" might seem like better approaches.

3.2 TextRank

TextRank[46] is a graph-based, unsupervised ranking algorithm for text processing, inspired by Google's PageRank[24]. While the algorithm can also be used for other NLP tasks, such as keyword/phrase extraction, this report will focus on its application to sentence extraction. Although the algorithm was first introduced in 2004, it is still considered very competitive and has been used as a benchmark for comparison in several other reports in the field[30, 38, 39].

The essence of the algorithm is to construct a graph $G = (V, E)$ and rank its vertices. For the application to sentence extraction for single document summarization, the graph G represents a text document that is to be summarized. Each vertex in V represents a sentence in the document. The edges in E represent relations between the vertices (sentences) in the graph. In difference to the sort of edges present in PageRank graphs, the edges in TextRank graphs carry weights. The weight on an edge between two vertices in the graph is determined by the similarity of the sentences that the vertices represent [46].

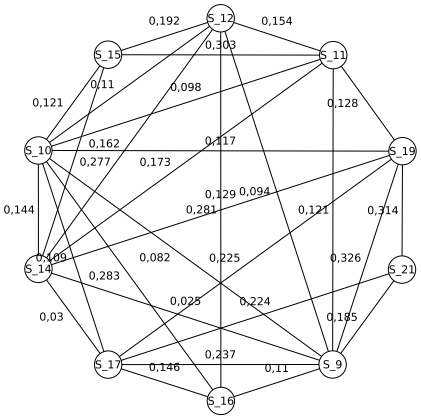


Figure 3.2: TextRank graph

The notion of sentence similarity can be measured in several different ways. The original paper on the algorithm presented results for one such similarity measure and suggested the use of several others. In this report we will describe, and present results for the use of TextRank with several different measurements of sentence similarity (including the one evaluated in the original paper).

When a graph has been constructed, the vertices are ranked according to the following formula:

$$Rank_{v_i} = (1-d) + d * \sum_{v_j \in In_{v_i}} \frac{w_{ji}}{\sum_{v_k \in Out_{v_j}} w_{jk}} Rank_{v_j}$$

Where $Rank_{v_i}$ stands for the rank of vertex v_i, d is a damping factor, In_{v_i} is the set of all vertices with edges going into vertex v_i, Out_{v_j} the set of all vertices which vertex v_j has an outgoing edge to and w_{ij} is the weight of the edge between vertices v_i and v_j. The damping factor d was set to 0.85, as suggested in the original paper[46]. This formula is a conversion of the PageRank algorithm, to work on graphs with weighted edges. The setting of these weights (by various similarity measures) will be addressed in section 4.2. The ranking step is repeated until convergence. Convergence is achieved when the error rate of any vertex in the graph is smaller than a given threshold. The error rate of a vertex is the change in that vertex's rank between two consecutive iterations. The error rate threshold used in this project was set to 0.0001.

3.3 K-means clustering

Using clustering for sentence extraction has been explored with good results[30]. The goal is to represent the sentences by some features that makes it possible to partition them into different clusters, where the sentences within a cluster are coherent to each other, but different to others. The sentences can then be selected from the clusters to extract a more diverse summary than TextRank. Partitioning entities into k different clusters is known as the K-means problem, and is defined as:

> Given a set $\{x_1, \ldots, x_n\}$, where each x_i is an m-dimensional vector, partition the n vectors into k partitions.

The algorithm can be divided into two steps, the first is the cluster assignment step. For each point x_i, find the closest cluster center, c_j:

$$assign(x_i, C_j) = min_{c_j} \parallel x_i - c_j \parallel^2$$

Where C_j is the cluster which cluster center, c_j, have the lowest euclidean distance to point x_i. The next step in the algorithm is the update of each clusters center value, according to the mean of point positions of the cluster:

$$\text{For each cluster } C_j,\ update(c_j) = \tfrac{1}{|C_j|} \textstyle\sum_{x_i \in C_j} x_i$$

This problem has been shown to be NP-complete[18], even for instances in the plane (i.e. $m = 2$) [44]. The original K-means clustering algorithm[42] solved the problem, but was shown to have a bad worst case, as well as resulting in arbitrarily bad clusters[19, 20]. An approximation called K-means++, which is $O(log\, k)$-competitive with the optimal clustering[20], was instead used as the main implementation for the clustering. The algorithm is an improvement over the original K-means algorithm by the addition of randomized seeding of the initial cluster centers.

3.3.1 Selection strategies

When the sentences have been clustered into k clusters, they need to be selected in some order to produce a summary of the desired length. To be able to produce a summary of a desired length of L words, the number of clusters k can be chosen such that $k = \frac{L}{avg_D}$, where avg_D is the average sentence length (in words) in document D. Selecting k sentences, one from each cluster, should give an approximate length of L words.

However, there are no guarantees that the resulting extraction is at most L words long. Sometimes the summary would exceed L words prior to selecting a sentence from each cluster, so that the last cluster(s) were not used to build the summary. From this came the need to order the clusters by importance, to ensure that sentences were selected from the more important clusters first. Consider 3 clusters ordered by descending importance as $\{B, A, C\}$, then the sentences should be selected in the order $\{b_1, a_1, c_1, b_2, a_2, c_2, \ldots\}$, where $a_i \in A$, $b_i \in B$, $c_i \in C$, until the summary reaches length L, or the clusters are depleted.

The first strategy that was implemented was to sort the clusters by size in descending order, making the largest cluster the most important. The intuition was that the largest cluster would concentrate on features that were common in many sentences, thereby making the "best" sentence of such a cluster representative of the document.

The second strategy was to consider the most dense cluster the most important, i.e. the cluster that has the least spread among its sentences. This was based on the intuition that such a cluster is well defined (because of the low spread) and the "best" sentence of such a cluster should thereby represent those features well.

Now that there's a strategy for the length of the summary, the problem is to choose in which order to select sentences from a cluster. A reasonable strategy is to pick the sentence from each cluster that is closest to its cluster center, with the intuition that such a sentence is the most representative of its cluster[30]. Two additional variants that seemed reasonable were also implemented. The first one was to order sentences in the same order they appeared in the document, since sentence position is often important in news corpora. The other one was to apply TextRank to the sentences of a cluster and then pick them in order of their vertex ranks.

The K-means algorithm provides no means of assigning real value ranks to the sentences of a document. Instead it returns the sentences as a list, ordered by relevance.

3.4 One-class SVM

Support Vector Machines (SVM) is a set of machine learning algorithms used in the task of data classification. The original task of a SVM is, given a set of data points, separate the points into two or more classes. For the case of linear two class classification, the goal of a SVM is to learn what hyperplane best separates the two classes.

The task of a one-class SVM is, given a set of data points all of the same class, learn the boundaries of that class in a feature space. What a one-class SVM requires in order to do this is a kernel function, $K(x, y)$, and the parameters ν and ϵ. $K(x, y)$, can be seen a as function for computing the similarity or distance between two data points, x and y in the input document. For the purpose of this project, $K(x, y)$ is applied to all pairs of sentence in a document, forming a $n \times n$ similarity matrix.

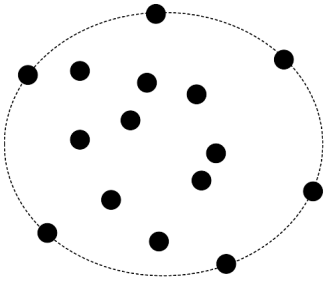


Figure 3.3: Support vectors on class boundary

In order to find the boundaries of a class, a one-class SVM identifies *support vectors*. A support vector is a class data point on the outline of the class. In figure 3.3 the support vectors corresponds to those points intersected by the dotted boundary line.

As mentioned in earlier sections, the graph based TextRank algorithm performs well in the task of automatic summarization. Recent work[33] have shown relations between identifying dense regions in a graph and solving a one-class SVM. The use of SVMs in clustering algorithms have also been evaluated with good results[23]. These observed relations to the two ranking approaches (TextRank and K-means clustering), presented in the previous sections spawned the idea of using one-class SVM for the purpose of sentence ranking.

Both two-class[32] and one-class[37] SVM have been explored in the field of sentence extraction before. However, these works are based on a supervised view, demanding the presence of a domain specific training set where sentences are marked as relevant or non relevant. We have not been able to find any previously published work exploring the use of one-class SVMs for automatic, extractive single document summarization in an unsupervised setting.

3.4.1 Sentence Ranking

The main question now is, how does one use this algorithm for sentence ranking? The SVM library LIBSVM[25] can train a one-class SVM giving as one of its results a set of support vectors. These vectors each corresponds to a sentence from the input set. Support vectors, in the SVM setting, is what separates a class from everything else. With this in mind the expectation is that the support vector sentences are important to a document and therefore may constitute a good summary.

As the support vector sentences are to form as the base of a summary, it is desirable to be able to control the number of support vectors. It has been shown that the parameter ν highly correlates

to the fraction of the data points that becomes support vectors[52]. From experiments it was clear that ν could control the number of support vectors, but could not be set statically as it is not predictable enough. More over, for the same ν, a different percentage of the sentences were selected as support vectors for different document. For the evaluation, the desired length measure of summaries are in numbers of words, not percentage of sentences. Therefore ν would have to be varied, even if it could give a predictable percentage. To be able to control the number of support vector sentences, ν was initialized to a low value and then increased until the support vectors could form a summary of the desired length. Formally:

```
staggerNu(K,ε,ν)
    solution = ocsvm(K,ε,ν)
    sents = getSupportVectors(solution)
    if(wordcount(sents)>d)
        return sents
    return staggerNu(K,ε,ν ∗ incFac)
```

where ocsvm(K,ϵ,ν) returns the solution of the one-class SVM, d is the desired summary length and $incFac$ is some constant > 1 (2 in this project). The parameter ϵ is set to 0.001 as suggested by the LIBSVM default settings.

Even though this iterative approach to set ν gives a set of sentences of the approximate length of a desired summary, it could return too many sentences. The sentences therefore need to be ranked, in order to construct the best summary. Each support vector is assign a coefficient value, α, by LIBSVM. These values are then used to rank the sentences, in decreasing order of α. As the values of α were found to be in a very narrow range, the one-class SVM algorithm simply returns the support vector sentences as a list ordered by relevance.

Chapter 4

Implementation

While chapter 3 describes the three sentence ranking algorithms in general, this chapter addresses further necessary steps needed to implement a full summarization system. What follows is a description of lexical preprocessing, different notions of sentence similarity, feature and kernel representations as well as methods for domain specific boosting and other means of result improvements.

4.1 Preprocessing

In order to summarize a document, using the different sentence ranking algorithms described in chapter 3, several preprocessing steps have to be executed. Each algorithm may require only certain preprocessing steps. Below all of these steps are summarized and discussed, in terms of the limitations they pose on the system as a whole.

4.1.1 Sentence splitting

To be able to view a document as a title paired with a list of sentences, one must first have a way to split the raw text representation of a document into sentences. Although this might sound like a trivial thing to do, naive approaches such as splitting on terminators (. ? ! etc.) give unsatisfactory results (consider, for example, abbreviations).

To address this problem, the Apache OpenNLP library was used in this project. The library contains pre-trained models for a number of text analysis tasks, for several languages. Thus given a language model and a input text, OpenNLP can split the text into sentences[11]. Even though the use of this library provides significantly better results than using pure, naive terminator splitting, the problem of faulty sentence splitting still remains to a certain extent. The need for a language specific model to be used with the library entails certain language dependency. Fortunately, models exist for the languages considered in this report (English and Swedish).

4.1.2 Tokenization

The algorithms used for sentence ranking in this project demand a more informative representation of a sentence than a pure text string. The sentences therefore need to be split up into words, to be used, for example, in sentence similarity comparison. Splitting a text string into words provides

several, language specific challenges. In English, for example, the use of apostrophes can be challenging for a computer to interpret[45].

Just as with sentence splitting, the Apache OpenNLP library was used in this project to address this challenge. The library provides functionality for splitting a input text string into words, given a language model[11]. Even though the tokenizer gives a good result, the problem of faulty tokenization is not entirely eliminated. The language limitations of the tokenizer is the same as that of sentence splitting, a language specific model needs to be provided.

4.1.3 Stemming

Several of the sentence ranking approaches presented in this thesis are centered around comparisons between words in different sentences. For example, if two sentences contain the same word, they can be considered to have a certain similarity. This means that a method to compare words is needed. A very naive way to do this is to just look at the word strings and see if they are equal. Such a solution, however, does not give a satisfying result, as different conjugations of the same word would be considered different words. To deal with this problem stemming was used. The software used to perform stemming in this project is a Java implementation of the Porter stemmer. The algorithm is rule based and is the most commonly used when it comes to stemming of English[45]. The stemmer is language dependent, however, it offers support for both languages considered here.

The process of stemming is built around removing word suffixes to reach the "root" form. However, different words can end up with the same stem. For example, the words *universe* and *university* would both likely be stemmed to *univers* and thus considered the same word. To address problems such as this, one could use more sophisticated techniques like lemmatization which uses morphological analysis to reach the "root" of words. The use of a lemmatizer might improve the performance of the system, but it would also make the system even more language dependent, as it requires more language knowledge than a stemmer[45].

At all points in this report where the concept of equal words is used, it means comparing the stemmed versions of the words.

4.1.4 Part of Speech (PoS) tagging

When considering what sentences to select from a text to form an extractive summary, words of certain lexical classes in those sentences can be more interesting than others. For example, the noun of a sentence is often central when it comes to the topic of the sentence. To be able to identify the classes of different words in a sentence, a PoS-tagger was used. The PoS-tagger takes as input a tokenized sentence and a language model and outputs the tag for each token in the sentence. The software used for this purpose was the Apache OpenNLP library[11] with language models for both Swedish and English.

Word comparison between sentences has a crucial role in most of the ranking approaches in this report. However, for several of the comparison approaches not all words in a sentence are considered. Sometimes it makes sense to only take words of certain PoS-classes into account when analyzing a sentence. For this purpose the notion of *PoS-filters* are used in this thesis. Applying a *PoS-filter* to a sentence means, to only consider words in the sentence with PoS-tags present in that *PoS-filter*. For instance, if a *PoS-filter* of nouns is applied to the sentence "Bob likes apples.", only the words "Bob" and "apples" are considered for comparison with other sentences.

4.1.5 Named Entity Recognition (NER)

One class of entities that are very important for some types of text, are names. If a sentence contains a name of a person, place, organization, etc., that name is probably very central to its context. To be able to use name analysis in the process of sentence extraction, NER software from the Stanford CoreNLP library was used[10]. This software is language dependent in the sense that it requires a language model for NER. Since no model for Swedish was found, this system could only be used for the English part of this project.

Also the Apache OpenNLP library[11] provides support for NER, but during a manual comparison of the systems for English, the Stanford library provided more favorable results. For the OpenNLP version, however, a Swedish language model was found, leading to the use of that system for Swedish texts.

4.1.6 Keyword extraction

A closely related task to that of extracting sentences from a text to form a summary, is that of extracting its most "important" words and/or phrases. In fact, one can view keyword extraction as the creation of an extractive summary, where the entities to extract are words (phrases) rather than sentences. Even though a list of keywords can be seen as a different way to summarize a document, it can also be seen as an aid in creating a sentence based summary. Knowing that a sentence contains a certain keyword can prove very useful when ranking the summary "worthiness" of that sentence.

In 2010 a master thesis on the subject of keyword extraction was carried out at Chalmers University of Technology and Findwise AB[34]. The thesis resulted in a system for identifying keywords/phrases in texts, which have also been used in this project. The system is language dependent, but has support for both languages evaluated in this report.

4.1.7 Term weighting by *tfidf*

The last three sections have been discussing preprocessing for supplying functionality to identify different kinds of "important" words. The notion of importance of words can also be achieved by weighting words (terms) differently. A common method to weigh a word is to check how "unusual" it is in a corpus compared to how often it occurs in the document currently processed.

A standard way to assign a weight to a word is to calculate its *tfidf* score, where *tf* stands for term frequency and *idf* for inverse document frequency. The *tf* of a term simply denotes the number of occurrences of the term in a specific document or sentence. To compute the *idf* of a term, one must have access to a collection (corpus) of documents to compute its document frequency (*df*), which denotes the number of documents containing the term. If one has access to the *df* of a term, its idf can be computed using the following formula[45]:

$$idf_t = log \frac{N}{df_t}$$

where idf_t is the *idf* of term t, N the number of documents in the corpus and df_t is the *df* of t. By combining the *tf* and *idf* of the t, the term can be weighted according to the following simple formula[45]:

$$tfidf_t = tf_t * idf_t$$

As different words are not equally common in different domains, computing the *df* of a term does not only require a corpus for a specific language, but also for a specific domain in the target language. The *df* of terms used in this project is based on the corpora of English and Swedish texts used for evaluation of the system.

4.1.8 Language dependency and limitations

The sentence ranking algorithms presented in this thesis are not, in themselves, language dependent. As they are unsupervised, they do not depend on language specific training sets. However, they do depend on the representation of sentences as features and the preprocessing needed to convert sentences into features is, as described above, highly language dependent.

Different versions of the ranking algorithms require different preprocessing. This means that the whole system is not dependent on the language support for every preprocessing step. All algorithms presented in this thesis require at the very least sentence splitting, tokenization and stemming to produce a summary. The absence of language models for the other preprocessing steps will only limit the options of the system.

4.2 Sentence similarity comparers

A sentence similarity comparer is, for the purpose of this report, simply a method which takes two sentences as input and gives a real number output measure of the similarity between the two sentences. The higher the outputted number is, the more similar the sentences are (according to some similarity measure). Sentence similarity is used in both TextRank graphs and one-class SVM kernels. Even though some of the comparers presented here allows for a similarity higher than 1 between two sentences, normalization of these values to weights in the interval $[0, 1]$ will be explored in section 4.5.3.

Several different similarity measures were tested for this report and each one forms the basis of a sentence similarity comparer. Below, the different sentence similarity comparers are described.

4.2.1 TextRank comparer

The original TextRank paper presents the following simple formula for measuring sentence similarity:

$$Similarity(S_1, S_2) = \frac{|S_1 \cap S_2|}{log|S_1| + log|S_2|}$$

This formula defines the similarity of two sentences (S_1 and S_2) simply as the number of shared words between them, divided by the sum of the *log* lengths of the sentences (in words). The paper also suggested the use of filtering of the words in the sentences, so that only words of certain lexical classes, such as nouns and verbs for instance, would be considered when counting the overlap of two sentences[46].

This similarity measure forms the basis for the implementation of a sentence similarity comparer called TextRank comparer. This comparer takes as input two sentences, which have been tokenized and PoS-tagged, as well as a PoS-filter. The comparer then applies the filter to the sentences and only considers the words passing through the filter:

$$Similarity(S_1,\, S_2) = \frac{|F(S_1 \cap S_2)|}{log|S_1| + log|S_2|}$$

where $F(S_1 \cap S_2)$ means applying PoS-filter F to words in the intersection between sentences S_1 and S_2.

4.2.2 *tfidf* comparer

In the TextRank comparer (described above), all words (that passes through the PoS-filter) are considered to be of equal worth. This means that if two sentences contains a common "usual" word, it contributes as much to their total similarity score as if they would share an "unusual" word. Although this may be fine for strict similarity measures, one can argue that this does not capture the fact that some words can be more important for certain documents than others. For example, a word that is very common in one document, but does not appear in many other documents, might be very important to that particular document. Sentences mentioning such an unusual word, can then be likely to be good candidates to use in the summary.

In order to try to make sentences that share important words recommend each other higher, a *tfidf* based version of the TextRank comparer was implemented. This *tfidf* comparer can be described with the following formula:

$$Similarity(S_1,\, S_2) = \frac{\sum_{w \in F(S_1 \cap S_2)} tf_w * idf_w}{log|S_1| + log|S_2|}$$

where tf_w is the term frequency for word w in the current document and idf_w the inverse document frequency for w in a corpus (in this project usually the corpus that the document belongs to).

4.2.3 Cosine comparer

As mentioned above, the original paper on TextRank suggested several other sentence similarity measures. One of those was cosine similarity[46].

Cosine similarity is a way to measure the similarity between two vectors by the difference of their angles. For vectors A and B, the cosine similarity between the vectors can be measured by the following formula[5]:

$$Similarity(A,\, B) = \frac{A \cdot B}{\parallel A \parallel \parallel B \parallel}$$

To be able to use cosine similarity as a measurement for sentence similarity, the first thing one have to do is to transform a sentence into a vector. This can be done by viewing each sentence as a N-dimensional vector, where N is the number of all unique words in the document which is to be summarized (the "target language"). Every dimension in such a "sentence vector" corresponds to a certain word, and if that word is present in the sentence, the value of that dimension in the vector is set to *tf* of the word in the sentence, multiplied by the *idf* of the word. For the dimensions in a sentence vector corresponding to words not present in the sentence, the value is set to 0. This gives rise to the following, *idf* weighted version of cosine similarity for two sentences S_1 and S_2:

$$Similarity(S_1,\, S_2) = \frac{\sum_{w \in F(S_1 \cap S_2)} tf_{w,S_1} tf_{w,S_2} (idf_w)^2}{\sqrt{\sum_{w \in F(S_1)} (tf_{w,S_1} idf_w)^2} * \sqrt{\sum_{w \in F(S_2)} (tf_{w,S_2} idf_w)^2}}$$

where $tf_{w,S}$ is the term frequency of word w in sentence S and idf_w is the inverse document frequency of word w[29]. In addition to this, the PoS-filtering idea described in the previous sections was also implemented for the cosine comparer.

4.2.4 Directed comparer

In the original paper on TextRank, the graphs that the algorithm is thought to work on have undirected edges[46]. This might seem quite natural as the edges represent sentence similarity and one might intuitively think that for sentences S_1 and S_2, $Similarity(S_1, S_2)$ and $Similarity(S_2, S_1)$ should be that same. Sentence S_1 should be as similar to sentence S_2 as sentence S_2 is to sentence S_1.

Logical as this seems, it does not capture the fact that a word can be described in more detail in one sentence than another. When it comes to using sentence extraction to create summaries, it seems quite natural, if faced with the choice of extracting two sentences describing the same (important) concept, to choose the more descriptive one. For example, consider the two sentences "A dog chased a cat."(S_1) and "A big angry dog chased a small scared cat."(S_2). These two sentences could describe the same event, but the S_2 gives the reader more information about the nature of the event and therefore might be a better choice for summary purposes.

To be able to capture this property of more descriptive sentences, the notion of directed graphs was introduced into the project. The idea is to let sentences like S_1 above recommend sentences like S_2 higher than sentences like S_2 recommend sentences like S_1. In figure 4.1, this corresponds to the weight $w_{1,2}$ having a larger value than $w_{2,1}$.

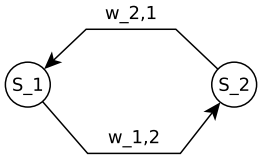


Figure 4.1: Nodes with directed edges

The words that take the most modifiers (such as adjectives, determiners, etc.) are often nouns, a noun with all its modifiers forms a noun phrase. When comparing the similarity between two sentences, using the Directed comparer, only the nouns in the sentences are considered. If two sentences share a common noun, the similarity contribution of that noun depend on the number of its modifiers. In this implementation all words in the phrase, except the currently examined noun are considered modifiers to that noun. This gives rise to the following formula for the noun contribution of the similarity between sentences S_1 and S_2:

$$NounContr(S_1, S_2) = \sum_{n \in S_1, S_2} modifiers_{s_2}(n)$$

where $modifiers_s(n)$ corresponds to the number of modifiers noun n has in sentence S. As $NounContr(S_1, S_2)$ is not necessarily equal to $NounContr(S_2, S_1)$ the weights of the two edges between the vertices have to be computed separately.

In order to identify noun phrases in a sentence, NP (Noun Phrase)-chunking library from the Apache OpenNLP[11] was used. The NP-chunker is language dependent in the sense that it requires a specific language model. As no Swedish model was found, the use of the Directed comparer is restricted to English texts only.

Another property that one can try to capture when representing a document as a directed graph, is references to a named entity by a pronoun in a later sentence. For instance, if a sentence, S_1, introduces a man by his name, say "Bob" and the next sentence, S_2 uses the word "he" or "him", without introducing a new name, it is probably also referring to "Bob". In a directed graph this can be captured by letting S_2 recommend S_1, without S_1 recommending S_2. More over:

```
NEAssos(S_1, S_2, t)
    pos = S_1.position - S_2.position;
    if(pos<t and pos>0 and S_1 contains named entity(ies) and
       S_2 does not contain named entity(ies), but contains pronoun(s))
        return 1;
    end if;
    return 0;
end NEAssos;
```

where t is a threshold for the distance between the sentences in the document.

The combination of the two ideas described above ($NounContr$ and $NEAssos$) forms the foundation for the Directed comparer. The comparer can be summarized by the following formula:

$$Similarity(S_1, S_2, t) = \frac{NounContr(S_1, S_2) + NEAssos(S_1, S_2, t)}{log|S_1| + log|S_2|}$$

4.2.5 Dependency Graph Kernel

The sentence similarity comparers presented so far have, more or less, viewed a sentence as a simple "bag of words" (bag of noun phrases in the case of the Directed Comparer). Another way to view a sentence is as dependency graph (or dependency tree), where the words in the sentence are the graphs vertices and the edges represents "functional relationships" between the words in a sentence. Each vertex in such a graph (tree), has exactly one parent, except for the root which has none[35]. For example the sentences "A fat cat was chased by a dog." and "A cat with a red collar was chased two days ago by a fat dog." can be represented by the following dependency graphs (trees):

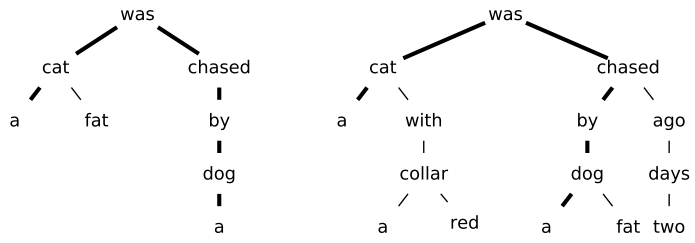


Figure 4.2: Dependency tree representation of sentences

To measure the similarity between two sentences, represented as in fig. 4.2, one can examine the common paths in their dependency trees[35]. In fig. 4.2, the longest common path between the two dependency trees is marked with thicker lines. In order to construct dependency structures as the ones described above, a dependency parsing system called MaltParser[15] was used in this project. The system is language independent in the sense that it can be trained for any language. However, no training was done since pre-trained models were available for English and Swedish. There is one major difference between the two language models. While the English model offers fast (linear) parsing, the Swedish one only offers slow (polynomial) parsing. Consequently only English texts have been evaluated using this similarity measure.

In [35] an algorithm to compute the similarity (called K below) between two dependency trees is given, built on what the author calls *CDP* (common downward paths) and *CPP* (common peak paths). The algorithm looks as follows (for a more detailed description see [35]):

$$
\begin{aligned}
K(T_1, T_2) &= \sum_{\substack{n_1 \in T_1 \\ n_2 \in T_2 \\ n_1.w = n_2.w}} (1 + CPP(n_1, n_2)) \\
CPP(n_1, n_2) &= CDP(n_1, n_2) + \sum_{\substack{c_1, \hat{c_1} \in C(n_1) \\ c_2, \hat{c_2} \in C(n_2) \\ c_1.w = c_2.w \\ \hat{c_1}.w = \hat{c_2}.w}} \begin{pmatrix} \alpha^2 + \alpha * CDP(c_1, c_2) + \\ \alpha * CDP(\hat{c_1}, \hat{c_2}) + \\ \alpha^2 * CDP(c_1, c_2) * CDP(\hat{c_1}, \hat{c_2}) \end{pmatrix} \\
CDP(n_1, n_2) &= \sum_{\substack{c_1 \in C(n_1) \\ c_2 \in C(n_2) \\ c_1.w = c_2.w}} (\alpha + \alpha * CDP(c_1, c_2))
\end{aligned}
$$

where T_i is the tree representation of sentence S_i, n_i is a node from tree T_i, $n_i.w$ is the word in node n_i and $C(n_i)$ denotes the child nodes of n_i. $CPP(n_i, n_j)$ denotes the "common peak path score" for all paths in the two trees which "peaks" at nodes n_i and n_j. The parameter $\alpha \in [0, 1]$ punishes long common paths, as these paths also include shorter common paths[35].

In addition to the this, the PoS-filtering idea was implemented such that a word $w \notin F(S)$ does not break a path, but does not add to the score. K is then normalized in the following way (as suggested in [35]) to form the Dependency Graph Kernel:

$$
Similarity(T_1,\, T_2) = \frac{K(T_1, T_2)}{\sqrt{K(T_1, T_1) * K(T_2, T_2)}}
$$

4.2.6 Filtered word comparer

When testing the Dependency Graph Kernel with TextRank on one of our corpora, some of the best results were observed with the parameter $\alpha = 0$. As this means that the paths in the dependency trees, does not contribute to the overall similarity measure between two sentences, the idea for a much simpler sentence similarity comparer was implemented. It counts common words (of certain word classes) and then normalizes according to the normalization step in the Dependency Graph Kernel. In more detail, the Filtered word comparer can be described with the following formula:

$$
Similarity(S_1,\, S_2) = \frac{|F(S_1 \cap S_2)|}{\sqrt{|F(S_1)| * |F(S_2)|}}
$$

4.2.7 Keyword comparer

In an attempt to only consider words with a high relevance to a document when computing sentence similarity, a similarity comparer based on document keywords was introduced. To be able to do this, a keyword extractor was used to extract a set of keywords, $\{kw_1, ..., kw_m\}$, for a document, where m is the desired number of keywords to extract. With the help of these keywords, each sentence, S_i, in the document can be represented as a m dimensional feature vector, $x_i = \{x_i^1, .., x_i^m\}$. Each dimension in the vector corresponds to a certain keyword in the set. The value for dimension x_i^j in x_i simply corresponds to the term frequency of keyword kw_j in S_i. To further limit the consideration of "unwanted" words for this comparison, the feature vector is also subject to a PoS-tag filter, so only keywords of certain PoS-tags are included as non zero values in the feature vector. Formally the construction of feature vector x_i can be expressed with the following formula:

$$x_i^{(j)} = |\{w | w \in F(S_i) \wedge w = kw_j\}|, \, for \, j = 1..m$$

When feature vectors have been constructed for all sentences in a document, the similarity of two sentences from that document can be computed using basic cosine similarity of their feature vectors:

$$Similarity(S_1, S_2) = \frac{x_1 \cdot x_2}{\| x_1 \| \| x_2 \|}$$

4.3 Sentences as feature vectors

To be able to cluster the sentences of a document, they need to be represented as feature vectors. Formally, a vector $x_i = \left[x_i^{(1)} \ \dots \ x_i^{(m)}\right] \in \mathbb{R}^m$ was constructed for each sentence S_i in the document D. Three different feature representations were implemented and are described in detail below.

4.3.1 Keyword features

A keyword extractor was used on the document to collect a set of m keywords $\{kw_1, \dots, kw_m\}$. The feature vector was then built by counting how many times each keyword kw_j occurs in the sentence S_i. A keyword is only counted if its PoS-tag is in the given filter. Formally, this is expressed as:

$$x_i^{(j)} = |\{w \,|\, w \in F(S_i) \wedge w = kw_j\}|, \text{ for } j = 1..m$$

The most common values are 0 or 1, since it is rare that a keyword is mentioned twice or more in the same sentence.

4.3.2 Filtered word features

A PoS-filter was used to collect a subset W containing m words from the document:

$$W = F(D), \, m = |W|$$

The feature vector was then built by, for each word $w_j \in W$, assign a value v_j to column j if the word is in the sentence. The value v_j can be configured to either be binary (0 or 1), or to be some term weighting of word w_j. Formally this is expressed as:

$$x_i^{(j)} = \begin{cases} v_j, & w_j \in S_i \wedge w_j \in W \\ 0, & otherwise \end{cases}, \text{ for } j = 1..m$$

In order to create feature representations similar to the *tfidf* and Cosine similarity comparers, the following term weighting was tested:

- *tfidf* with term frequency on document level (*tfidf* comparer)
- *tfidf* with term frequency on sentence level (Cosine comparer)

To be able to further mimic the similarity comparers, the feature values were normalized in the following ways:

- Log sentence length : $Normalize(v_j) = \frac{v_j}{log(|S_i|)}$ (*tfidf* and TextRank comparers)
- Vector length: $Normalize(v_j) = \frac{v_j}{||x_i||}$ (Cosine comparer)

4.3.3 NER features

A named entity extractor was used on the document to collect a set of m named entities $\{e_1, \ldots, e_m\}$. The feature vectors were then composed by counting the occurrences of named entities in the sentence. A named entity may consist of several tokens (e.g. a first name and surname), thus every e_j is a list of tokens. The counting was implemented such that a word w is counted each time it is part of a named entity. Consider for instance that some e_j contains 3 tokens and that sentence S_i contains all of them, then $x_i^{(j)} = 3$, which implies that S_i is well connected to the named entity e_j, whereas if some other sentence S_t only contains 1 of the tokens, then $x_t^{(j)} = 1$ which implies that it is only somewhat connected to e_j. Formally, building the vectors can be expressed as:

$$x_i^{(j)} = |\{w \,|\, w \in S_i \wedge w \in e_j\}|, \text{ for } j = 1..m$$

4.4 Sentence similarity as kernel functions

The similarity measures for sentences described in section 4.2 can be used as kernel functions for the one-class SVM. In fact, one can even go as far as to translate a TextRank graph, $G = (V, E)$, directly into a SVM kernel matrix in the following way:

$$\begin{aligned} W_{ij} &= \begin{cases} 0 & e_{ij} \notin E \\ e_{ij}.w & otherwise \end{cases} \\ K &= W + I \end{aligned}$$

where e_{ij} is the edge between vertices v_i and v_j in V, $e_{ij}.w$ the weight of edge e_{ij}, K the kernel matrix, W the weight matrix of the edges in E and I the identity matrix. As K can be seen

as a similarity matrix, only sentence similarity comparers producing symmetric weight matrices ($Similarity(S_i, S_j) = Similarity(S_j, S_i)$) can be used here, meaning the Directed Comparer cannot be applied. The reason for the addition with I is simply to capture that the "similarity" between a sentence and itself is always 1 (maximum). As SVMs works best on normalized kernel matrices ($[0, 1]$ or $[-1, 1]$), the values of W are normalized in the range $[0, 1]$ (see section 4.5.1).

The authors of [33] propose the use of another translation between a graph, $G = (V, E)$,and a kernel matrix:

$$\begin{aligned} K &= \frac{A}{-\lambda_{min}(A) * (1 + \delta)} + I \\ A_{ij} &= \begin{cases} 1 & e_{ij} \in E \\ 0 & otherwise \end{cases} \end{aligned}$$

where A is the adjacency matrix of the the graph, $\lambda_{min}(A)$ is the minimum eigenvalue of A and δ is some constant > 0. The use of the adjacency matrix of the graph suites unweighted graphs better than the weighted graphs used in TextRank. This have given rise to the following two variations of the formula:

$$\begin{aligned} K_1 &= \frac{W}{-\lambda_{min}(W) * (1 + \delta)} + I \\ K_2 &= \frac{A'}{-\lambda_{min}(A') * (1 + \delta)} + I \\ A'_{ij} &= \begin{cases} 1 & W_{ij} > t \\ 0 & otherwise \end{cases} \end{aligned}$$

where t is some predefined threshold > 0. For kernel representation K_2, the choice of t is important. As the weights in W can vary significantly between different comparers, and even different documents, it is unfeasible to set it statically. Instead a statistical approach was taken, where both the mean and the median of the weights of the edges in E were considered as thresholds.

4.5 Boosting and improvements

The three ranking algorithms are designed to be, more or less, independent of the type of text that is to be summarized. While this can be viewed a strength, it is possible to add features on top of them to improve the quality of the summaries in certain domains (using some domain knowledge). As the corpus most widely used for evaluation in this report (DUC2002) consists of short news articles, the boosting strategies in this part were developed to boost results in that domain.

4.5.1 Edge normalization

As an experimental attempt to improve the system performance, edge normalization was implemented. The weights of the edges in the TextRank graphs and one-class SVM kernel matrices were normalized to be in the range of $[0, 1]$ using the following formula based on Laplacian matrix normalization[6]:

$$Normalized(w_{v_i v_j}) = \frac{w_{v_i v_j}}{\sqrt{\sum_{l=1}^{n} w_{v_i v_l} * \sum_{m=1}^{n} w_{v_j v_m}}}$$

where $w_{v_i v_j}$ is the weight of the edge between vertex v_i and vertex v_j (similarity between sentences S_i and S_j) and n is the total number of vertices in the graph.

As this normalization approach relies on the graph being undirected ($w_{v_i v_j} = w_{v_j v_i}$), the Directed Comparer can not be normalized this way. This comparer is instead normalized using the following, naive formula:

$$Normalized(w_{v_i v_j}) = \frac{w_{v_i v_j}}{w_{max}}$$

where w_{max} is the maximum edge weight in the graph.

4.5.2 Eliminating short sentences

One can argue that sentences shorter than a certain length are unlikely to supply enough information to be considered useful for an extractive summary. With this in mind only sentences of a certain input length (in words) and higher are considered for inclusion in a summary.

4.5.3 Position analysis

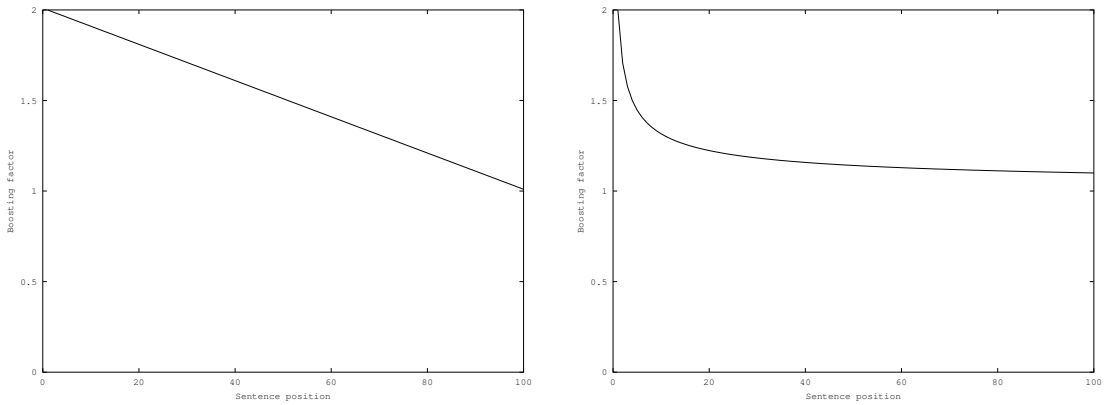
In texts such as news articles, the first sentences usually describe the major content of the text. This means sentences in the beginning of news articles are more probable to be useful in the construction of the summary than other sentences in the text.

Position based rank boosting

To be able to take advantage of position importance one can, after ranking all sentences in a text, boost the ranks of the sentences according to their position. For this purpose, we propose the following boosting algorithms:

$$\begin{aligned} Boost_a(S_i) &= S_i.rank * \left(1 + \frac{|S| + 1 - S_i.pos}{|S|}\right) \\ Boost_b(S_i) &= S_i.rank * \left(1 + \frac{1}{\sqrt{S_i.pos}}\right) \end{aligned}$$

where S is the set of sentences in a document, $S_i.rank$ is the rank of sentence S_i and $S_i.pos$ is the original position of S_i in the document (starting at 1). Version b yielded higher ROUGE scores for TextRank on the DUC2002 corpus. The curves displayed in fig. 4.3 show that $Boost_b$ gives only the first few sentences a high boost, which fits the structure of news articles as mentioned earlier. $Boost_a$ however is linearly decreasing and might give too much contribution to sentences outside the introduction part of an article. Consequently, $Boost_b$ will be the method referred to as *position boosting*. This sort of position based boosting is only applicable to algorithms giving each sentence a real value rank, not to algorithms returning a relevance ordering of sentences.



(a) Linear (b) Inverse square

Figure 4.3: Variations of position boosting

A seeder based on sentence position

For the K-means algorithm, another way to exploit the position relevance was tested. Clustering sentences into k clusters requires K-means++ to choose k initial cluster centers, which is called seeding. The default seeder for K-means++ is based on randomization. The position seeder on the other hand is deterministic and simply picks the first k sentences, as they appear in the document, as the initial cluster centers. This strategy works reasonably well for a news corpus, since the initial sentences often contain a lot of information while remaining diverse.

4.5.4 Title analysis

The title of a document such as a news article can, arguably, be seen as a very short summary of the document. With this in mind, sentences that are describing topics mentioned in the title might seem like viable candidates to select for an extractive summary. To test this idea, the rank of sentences sharing one or more named entities with the title was boosted:

$$Boost(S_i) = S_i.rank * \begin{cases} w, & S_i\, contains\, N.E.\, from\, title \\ 1, & otherwise \end{cases}$$

where w is a boosting factor > 1.

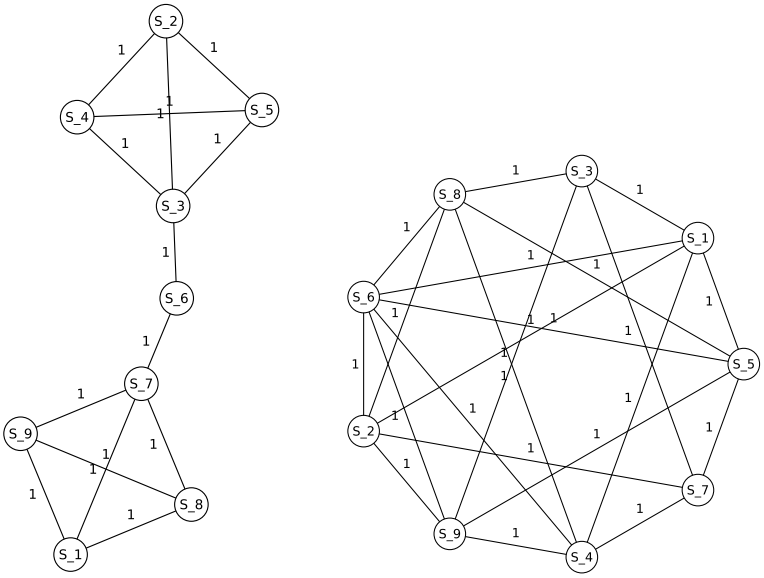
This boosting technique can only be use for algorithms providing a real value rank for each sentence.

4.5.5 Inverted adjacency and weight matrices

For the one-class SVM algorithm, inverted versions of A' and W were evaluated. When inverting the adjacency matrix of a graph, the dense parts become sparse and vice versa. Formally, a matrix M can be inverted to $\widetilde{M}$ in the following way:

$$\widetilde{M_{ij}} = \begin{cases} 0 & i = j \\ 1 - M_{ij} & otherwise \end{cases}$$

If the graph contains any cliques, they will become independent sets and vice versa. Since similar sentences are connected in the normal graph, they will become disconnected in the inverted graph and instead connected to sentences addressing other topics. This yields a very different graph and seemed like an interesting structure to explore. An example is visualized in fig. 4.4.



(a) Normal graph with adjacency matrix A

(b) Inverted graph with adjacency matrix $\widetilde{A}$

Figure 4.4: Graph inversion

Chapter 5

Experiments

In this chapter results for experiments conducted to measure the performance of the three sentence ranking algorithms will be presented. For each algorithm results for summary generation in both English and Swedish will be presented. For evaluation in English the DUC2002 corpus is used, while the system is evaluated on the Wikipedia corpus to measure the performance for Swedish texts. For the DUC2002 corpus the target length of a summary is 100 words. As the abstracts for the Swedish corpus vary significantly in length, the target length of the summaries are here based on the length of the gold standard abstracts as mentioned earlier. In all experiments conducted in this chapter, the minimum length of sentences to be considered for extraction is 8 words. In the tables below, different abbreviations are used to represent different word classes (PoS-filters). N stands for nouns, Adj for adjectives, AV for adverbs and V for verbs. The results presented in the tables are those of the highest observed ROUGE Ngram(1,1) score for each variation of each algorithm.

5.1 TextRank Results and discussion

In the experiments conducted on the TextRank algorithm, all edge weights in the graphs are normalized according to the methods described in section 4.5.3. All sentence similarity comparers presented in section 4.2 are evaluated for the English corpus, while language dependency limits which of these can be tested for the Swedish corpus. Both position and title similarity based boosting is explored for the evaluation of English texts, while only position boosting is tested for Swedish (the Wikipedia corpus provides no document titles).

5.1.1 English corpus

Similarity Comparer	PoS-filter	Recall	Precision	F_1
TextRank Comparer	N, Adj	0.4558	0.4438	0.4483
tfidf Comparer	N, Adj	**0.4577**	**0.4494**	**0.4523**
Cosine Comparer	-	0.4478	0.4403	0.4427
Directed Comparer	-	0.4515	0.4415	0.4450
Dep. Graph Kernel ($\alpha = 0.2$)	N, Adj	0.4300	0.4165	0.4217
Filtered word Comparer	N, Adj	0.4426	0.4381	0.4390
Keyword Comparer ($N = 20$)	N, Adj	0.4431	0.4384	0.4395

(a) No boosting

Similarity Comparer	PoS-filter	Recall	Precision	F_1
TextRank Comparer	N	0.4723	0.4617	0.4655
tfidf Comparer	N, Adj, V, AV	**0.4793**	**0.4706**	**0.4736**
Cosine Comparer	-	0.4743	0.4681	0.4699
Directed Comparer	-	0.4653	0.4555	0.4588
Dep. Graph Kernel ($\alpha = 0.2$)	N, Adj, V, AV	0.46919	0.46173	0.4641
Filtered word Comparer	N, Adj	0.4704	0.4660	0.4669
Keyword Comparer ($N = 20$)	N, Adj, V, AV	0.4680	0.4619	0.4637

(b) Position boosting

Similarity Comparer	PoS-filter	Recall	Precision	F_1
TextRank Comparer	N, Adj	0.4542	0.4426	0.4469
tfidf Comparer	N, Adj	**0.4580**	0.4479	**0.4516**
Cosine Comparer	-	0.4471	0.4394	0.4419
Directed Comparer	-	0.4529	0.4418	0.4458
Dep. Graph Kernel ($\alpha = 0.2$)	N, Adj	0.43368	0.42053	0.42549
Filtered word Comparer	N, Adj	0.4430	0.4373	0.4388
Keyword Comparer ($N = 20$)	N, Adj	0.4431	**0.4553**	0.4379

(c) Title boosting ($w = 1.5$)

Table 5.1: TextRank similarity comparer results on the DUC2002 corpus

Table 5.1 shows that using the *tfidf* Comparer as similarity comparer in TextRank yields the highest ROUGE score in all three categories of boosting (unboosted, position, title). It is disappointing that the more NLP oriented comparers (e.g. Dependency Graph Kernel) are outperformed by this rather naive approach.

By only viewing these results, one can argue that the choice of similarity comparer is not that crucial for TextRank. The difference in F_1 score between the highest and the lowest is merely 0.013. However, a larger gap was observed before introducing some of the ideas from section 4.5.3 (most notably eliminating short sentences).

In terms of boosting, the position boosting improves the F_1 scores significantly, while the title boosting does not improve it. The assumed reason that the position boosting works is, as mentioned in 4.5.3, the general structure of news articles. A reason to why the title boosting works poorly might be because the titles of news articles are often designed to capture the interest of a reader rather than give a good summary of its content. This could also lead to a title which contains no

named entities, making the boosting factor the same for each sentence. Consider for instance this title:

Reportage On Provincial Flood Situation

The article describes flooding events in three Chinese provinces (Guangdong, Hunan and Jiangxi). Even though the title describes flooding, it does not say anything specific about the event, and also contains no named entities. A better title booster could probably be implemented to leverage the titles of news articles.

5.1.2 Swedish corpus

Similarity Comparer	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank Comparer	N, Adj	0.3689	0.3304	0.3473
TFIDF Comparer	N, Adj	0.3556	0.3247	0.3384
Cosine Comparer	-	0.3666	0.3342	0.3487
Filtered word Comparer	N, Adj	**0.3744**	**0.3360**	**0.3531**
Keyword Comparer	N, Adj	0.3670	0.3326	0.3481

(a) No boosting

Similarity Comparer	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank Comparer	N, Adj	0.3752	0.3421	0.3569
TFIDF Comparer	N, Adj	0.3698	0.3394	0.3530
Cosine Comparer	-	0.3762	**0.3445**	0.3588
Filtered word Comparer	N, Adj	**0.3783**	0.3438	**0.3593**
Keyword Comparer	N, Adj	0.3699	0.3384	0.3526

(b) Position boosting

Table 5.2: TextRank similarity comparer results on the Wikipedia corpus

In contrast to the English corpus, the Filtered word comparer seems to outperform the others here. Although, all the comparers perform quite equal when boosted, the Filtered word comparer also performs strongly unboosted. The gap between the best and worst performance is (in $F_1 - score$) 0.0147 unboosted and 0.0067 when boosted.

An interesting thing to notice for this corpus is that position boosting does not improve the scores as much as for the DUC2002 corpus. The increase in $F_1 - score$ gained by boosting the Filtered word comparer, for instance, is a mere 0.0062, compared to 0.0279 for the English corpus. This can probably be explained by the domains of the corpora. The DUC2002 corpus consists of very short articles (29 sentences in average), while the articles in the Swedish Wikipedia corpus are quite long (200 sentences in average). The Wikipedia articles are divided into several sub paragraphs. Therefore a position boosting relative to a paragraph, rather then the full text may perform better in this domain.

5.2 K-means Results and discussion

For the evaluation of the K-means algorithm for the English corpus, all feature representations described in section 4.3 are tested. When it comes to Swedish, only the keyword and named entities representations are explored due to the extensive length of the filtered word feature vectors in this corpus.

5.2.1 English corpus

Features	PoS-filter	Term weighting	Normalization	Recall	Precision	F_1
Keywords($N = 23$)	N, Adj	-	-	0.4298	**0.4327**	**0.4303**
Filtered words$_1$	N, Adj, AV, V	-	Log sent. length	0.3979	0.4081	0.4017
Filtered words$_2$	N	*tfidf* (doc. level)	Log sent. length	0.3685	0.3775	0.3720
Filtered words$_3$	N, Adj, AV, V	*tfidf* (sent. level)	Vector length	0.4144	0.4204	0.4164
Filtered words (Best)	N, Adj, AV, V	-	Vector length	0.4239	0.4265	0.4240
Named entities	-	-	-	**0.4431**	0.3968	0.418

(a) Centroid based in cluster ranking

Features	PoS-filter	Term weighting	Normalization	Recall	Precision	F_1
Keywords ($N = 15$)	N, Adj, AV, V	-	-	0.4668	0.4598	0.4621
Filtered words$_1$	N, Adj, AV, V	-	Log sent. length	0.4581	0.4435	0.4492
Filtered words$_2$	N, Adj	*tfidf* (doc. level)	Log sent. length	0.4659	**0.4639**	**0.4636**
Filtered words$_3$	N, Adj, AV, V	*tfidf* (sent. level)	Vector length	0.4544	0.4521	0.4521
Named entities	-	-	-	**0.4755**	0.4227	0.4463

(b) Position based in cluster ranking

Features	PoS-filter	Term weighting	Normalization	Comparer	Recall	Precision	F_1
Keywords ($N = 22$)	N, Adj	-	-	*tfidf*(*)	0.4720	**0.4586**	**0.4640**
Filtered words$_1$	N, Adj, AV, V	-	Log sent. length	*tfidf*(*)	0.4633	0.4483	0.4541
Filtered words$_2$	N	*tfidf* (doc. level)	Log sent. length	*tfidf*(*)	0.4677	0.4619	0.4635
Filtered words$_3$	N, Adj	*tfidf* (sent. level)	Vector length	*tfidf*(*)	0.4601	0.4537	0.4556
Named entities	-	-	-	*tfidf*(*)	**0.4788**	0.4228	0.4477

(c) TextRank used for in cluster ranking

Table 5.3: Using K-means++ seeding

Features	PoS-filter	Term weighting	Normalization	Recall	Precision	F_1
Keywords ($N = 16$)	N	-	-	0.4462	0.4445	0.4440
Filtered words$_1$	N, Adj	-	Log sent. length	0.4271	0.4362	0.4307
Filtered words$_2$	N	*tfidf* (doc. level)	Log sent. length	0.3924	0.3959	0.3931
Filtered words$_3$	N, Adj, AV, V	*tfidf* (sent. level)	Vector length	0.4435	0.4425	0.4418
Filtered words (Best)	N, Adj, AV, V	-	Vector length	0.4471	**0.4455**	**0.4450**
Named entities	-	-	-	**0.4660**	0.4184	0.4399

(a) Centroid based in cluster ranking

Features	PoS-filter	Term weighting	Normalization	Recall	Precision	F_1
Keywords ($N = 15$)	N	-	-	0.4689	0.4608	**0.4634**
Filtered words$_1$	N, Adj, AV, V	-	Log sent. length	0.4624	0.4568	0.4583
Filtered words$_2$	N	*tfidf* (doc. level)	Log sent. length	0.4593	0.4573	0.4572
Filtered words$_3$	N	*tfidf* (sent. level)	Vector length	0.4600	0.4603	0.4590
Filtered words (Best)	N, Adj	-	Vector length	0.4613	**0.4612**	0.4602
Named entities	-	-	-	**0.4860**	0.4342	0.4575

(b) Position based in cluster ranking

Features	PoS-filter	Term weighting	Normalization	Comparer	Recall	Precision	F_1
Keywords ($N = 18$)	N, Adj	-	-	*tfidf*(*)	0.4739	0.4635	0.4672
Filtered words$_1$	N	-	Log sent. length	*tfidf*(*)	0.4726	0.4645	0.4673
Filtered words$_2$	N	*tfidf* (doc. level)	Log sent. length	*tfidf*(*)	0.4630	0.4569	0.4588
Filtered words$_3$	N, Adj, AV, V	*tfidf* (sent. level)	Vector length	*tfidf*(*)	0.4669	**0.4650**	0.4646
Filtered words (Best)	N, Adj	-	-	*tfidf*(*)	0.4747	0.4641	**0.4680**
Named entities	-	-	-	*tfidf*(*)	**0.4900**	0.4364	0.4605

(c) TextRank used for in cluster ranking

Table 5.4: Using position seeding

In tables 5.3c and 5.4c *tfidf*(*) refers to the *tfidf* comparer with PoS-filter {N, Adj, AV, V} and position boosting. Filtered words$_{1-3}$ are combinations of term weighting/normalization that corresponds to the similarity comparers in TextRank. Filtered words$_1$ corresponds to the TextRank comparer, Filtered words$_2$ to the *tfidf* comparer and Filtered words$_3$ to the cosine comparer. If any other combination beats these three, it will be included in the results marked Filtered words (Best).

From the results in table 5.3 and 5.4 it is clear that all three variables (feature representation, initial seeding and in cluster ranking) play a big roll in the performance of the algorithm. The best results are obtained with very position dependent seeding and in cluster ranking. As the importance of sentence position in a document is very domain specific, it is hard to say how well these seedings and rankings perform in other domains. The settings used in table 5.3a are the most domain independent, but gives the lowest score. The settings giving the best result, found in table 5.4c, is position dependent both in initial seeding and in cluster ranking.

When it comes to feature representation, they all seem to perform good in combination with some seeding/in cluster ranking. The keyword and filtered words features both yields high precision and $F_1 - score$, while the named entity representation yields the highest recall.

5.2.2 Swedish corpus

Features	PoS-filter	Recall	Precision	F_1
Keywords ($N = 19$)	Adj, N	**0.3566**	**0.3265**	**0.3402**
NER		0.3407	0.3160	0.3268

(a) Centroid based in cluster ranking

Features	PoS-filter	Recall	Precision	F_1
Keywords ($N = 18$)	V, AV, Adj, N	**0.3624**	**0.3347**	**0.3473**
NER		0.3424	0.3209	0.3302

(b) Position based in cluster ranking

Features	PoS-filter	Comparer	Recall	Precision	F_1
Keywords ($N = 19$)	V, AV, Adj, N	Filtered word	**0.3600**	**0.3403**	**0.3539**
NER			0.3459	0.3219	0.3322

(c) TextRank used for in cluster ranking

Table 5.5: Using K-means++ seeding

Features	PoS-filter	Recall	Precision	F_1
Keywords ($N = 18$)	V, AV, Adj, N	**0.3542**	0.3246	**0.3380**
NER		0.3480	**0.3249**	0.3351

(a) Centroid based in cluster ranking

Features	PoS-filter	Recall	Precision	F_1
Keywords ($N = 19$)	Adj, N	**0.3575**	**0.3283**	**0.3413**
NER		0.3497	0.3247	0.3356

(b) Position based in cluster ranking

Features	PoS-filter	Comparer	Recall	Precision	F_1
Keywords ($N = 19$)	Adj, N	Filtered word	**0.3616**	**0.3345**	**0.3467**
NER			0.3524	0.3273	0.3382

(c) TextRank used for in cluster ranking

Table 5.6: Using position seeding

For the two feature representations evaluated for this corpus, it is clear that the keywords outperforms the named entities. An interesting point is that the k-means++ seeding performs better than the position seeding for this corpus, which can be connected to the points made earlier about position boosting of TextRank for this corpus.

5.2.3 Observations

During testing it became clear that the clustering of a document using this algorithm produced clusters of quite different sizes. More over, it was often observed that a few of the clusters for a

document only contained one sentence. Doing in cluster ranking on a cluster with only one data point is, of course, irrelevant. This means that if a cluster only contains one sentence, that sentence will always be selected to be in the final summary (if the cluster is at all considered for extraction).

Another observation made was that long feature vectors slows down the algorithm significantly. Feature representations providing lengths in the magnitude of the number of unique words in a document may therefore be insufficient for longer texts (such as those in the Wikipedia corpus).

5.3 One-class SVM results and discussion

For the evaluation of the one-class SVM algorithm in English, all undirected similarity comparers described in section 4.2 are applied to form kernel matrices using both representations, K_1 and K_2 (described in section 4.4). Language dependencies limits which similarity comparers are tested for the Swedish corpus. Both normal and inverted versions of the kernels are tested.

5.3.1 English corpus

Comparer	**PoS-filter**	**Recall**	**Precision**	$F_1 - score$
TextRank Comparer	N., Adj., AV., V.	**0.39464**	**0.40917**	**0.40086**
tfidf Comparer	N., V.	0.38802	0.40136	0.39372
Cosine Comparer	N., Adj., AV., V.	0.39338	0.40614	0.39881
Dep. Graph Kernel ($\alpha = 0.2$)	N., V.	0.39219	0.40823	0.39899
Filtered word Comparer	N., Adj., AV., V.	0.39569	0.4066	0.40014
Keyword Comparer($N = 20$)	N., Adj., AV., V.	0.37128	0.3832	0.37617

(a) Kernel representation K_1

Comparer	**PoS-filter**	**Recall**	**Precision**	$F_1 - score$
TextRank Comparer	N., V.	0.39044	0.40231	0.3953
tfidf Comparer	N.	0.38204	0.39251	0.38636
Cosine Comparer	N., Adj.	0.38807	0.39525	0.39068
Dep. Graph Kernel ($\alpha = 0.2$)	N., Adj, AV., V	**0.39722**	**0.4118**	**0.40343**
Filtered word Comparer	N., Adj, AV., V	0.39585	0.40446	0.39907
Keyword Comparer($N = 20$)	N., V.	0.3844	0.39315	0.38769

(b) Kernel representation K_2

Table 5.7: Normal weight matrix representation

Comparer	**PoS-filter**	**Recall**	**Precision**	$F_1 - score$
TextRank Comparer	N., Adj.	0.4276	0.4237	0.4243
tfidf Comparer	N., Adj., AV., V	**0.4391**	**0.4306**	**0.4334**
Cosine Comparer	N., V.	0.4219	0.4217	0.4204
Dep. Graph Kernel ($\alpha = 0.2$)	N., Adj., AV., V	0.4196	0.4219	0.4195
Filtered word Comparer	N., Adj	0.4195	0.4203	0.4187
Keyword Comparer($N = 20$)	N.	0.4226	0.4232	0.4218

(a) Kernel representation K_1

Comparer	**PoS-filter**	**Recall**	**Precision**	$F_1 - score$
TextRank Comparer	N., Adj.	0.4236	0.4224	0.4220
tfidf Comparer	N.	**0.4279**	**0.4271**	**0.4264**
Cosine Comparer	N., Adj., AV., V	0.4217	0.4230	0.4213
Dep. Graph Kernel ($\alpha = 0.2$)	N., Adj., AV., V	0.4074	0.4067	0.4060
Filtered word Comparer	N., Adj.	0.4168	0.4202	0.4174
Keyword Comparer($N = 20$)	N., Adj.	0.4184	0.4187	0.4175

(b) Kernel representation K_2

Table 5.8: Inverted weight matrix representation

From the results in tables 5.7 and 5.8, it is clear that inverting the weight matrices in the kernels improves the ROUGE-score for this corpus. When it comes to the weighted (K_1) versus the adjacency (K_2) version of the kernel matrices, they give quite even scores. K_1 performs slightly better on the inverted version, while K_2 is slightly better on the normal version of the weight matrix. However, the difference is too small to say anything about which may be the best representation in the general case.

5.3.2 Swedish corpus

Comparer	PoS-filter	Recall	Precision	$F_1 - score$
TextRank Comparer	N, Adj, AV, V	0.3256	0.3054	0.3146
tfidf Comparer	N, Adj, AV, V	0.3253	0.3042	0.3139
Cosine Comparer	N, Adj, AV, V	0.3264	0.3042	0.3143
Filtered word Comparer	N, Adj, AV, V	**0.3266**	**0.3062**	**0.3155**
Keyword Comparer($N = 20$)	N, Adj, AV, V	0.3232	0.2998	0.3103

(a) Kernel representation K_1

Comparer	PoS-filter	Recall	Precision	$F_1 - score$
TextRank Comparer	N, Adj	**0.3417**	**0.3190**	**0.3292**
tfidf Comparer	N, Adj	0.3362	0.3142	0.3242
Cosine Comparer	N, Adj	0.3365	0.3146	0.3245
Filtered word Comparer	N, Adj	0.3403	0.3186	0.3284
Keyword Comparer($N = 20$)	N, V	0.3348	0.3119	0.3222

(b) Kernel representation K_2

Table 5.9: Normal weight matrix representation

Comparer	PoS-filter	Recall	Precision	$F_1 - score$
TextRank Comparer	N, Adj, AV, V	0.3358	0.3104	0.3219
tfidf Comparer	N, V	**0.3458**	0.3184	**0.3308**
Cosine Comparer	N, Adj, AV, V	0.3452	**0.3189**	0.3307
Filtered word Comparer	N, Adj	0.3404	0.3120	0.3248
Keyword Comparer($N = 20$)	N, V	0.3427	0.3160	0.3280

(a) Kernel representation K_1

Comparer	PoS-filter	Recall	Precision	$F_1 - score$
TextRank Comparer	N, Adj, AV, V	0.3433	0.3168	0.3288
tfidf Comparer	N, Adj, AV, V	0.3485	0.3213	0.3336
Cosine Comparer	N, V	**0.3554**	**0.3270**	**0.3399**
Filtered word Comparer	N, Adj, AV, V	0.3393	0.3165	0.3269
Keyword Comparer($N = 20$)	N, V	0.3421	0.3156	0.3275

(b) Kernel representation K_2

Table 5.10: Inverted weight matrix representation

For this corpus kernel representation K_2 outperforms K_1 for both normal and inverted graphs. The highest F_1 score is once again observed for inverted weight matrices for both K_1 and K_2.

5.3.3 Observations

When testing the one-class SVM sentence ranking, some alarming observations were made. First, the value of ν for the desired summary length was significantly lower than expected. Previous

work, stating the correlation of ν and the fraction of support vectors, suggest values of ν in the range $[0.01, 0.5]$ which should correspond to fractions in the range $[10\%, 51.2\%]$[52]. However, in the evaluation of this system the average desired fraction was 30%, but to achieve this, ν had to be of the magnitude 0.0001.

Another observation was that the first sentence was selected very often. An experiment was conducted to analyze this behavior where summaries were produced for 100 randomly chosen documents. The sentences were randomly shuffled before building the kernel matrix for a document. The experiment showed that in the basic setting with a weighted matrix W, and no filtering of short sentences, the first sentence was selected in 94/100 summaries. This number decreased if the adjacency matrix A was used instead, and further if the matrix was inverted and short sentences were filtered. With this setting, the first sentence was selected in 62/100 summaries. This can be compared to the basic setting of TextRank, still using shuffled sentences, where the first sentence was selected in 18/100 summaries.

With these observations in mind, further testing and investigation should probably be carried out before using the one-class SVM approach on a larger scale.

5.4 Testing on equal terms

Even though the experiments conducted on TextRank and one-class SVM uses the same measures of sentence similarity, the same cannot be said for K-means clustering. The objective of the K-means algorithm is to, in each iteration, assign all points to the nearest cluster centroid. This task can be rephrased as to finding the cluster centroid most similar to each point. When thinking of the task as a similarity comparison between points and cluster centroids, it is possible to construct a similarity measure based on the K-means algorithm.

Formally the assignment step of K-means can be expressed as, for each point, x_i, find the cluster center, c_j, that minimizes:

$$\| x_i - c_j \|^2 = x_i^T \cdot x_i + c_j^T \cdot c_j - 2x_i^T \cdot c_j$$

It is clear from the formula above that minimizing $\| x_i - c_j \|^2$ is equivalent to maximizing $x_i^T \cdot c_j$. With this in mind one can argue that $x_i^T \cdot c_j$ can be seen as a similarity measure between point x_i and center c_i, the higher the similarity, the closer the point is to the center.

To mimic the similarity measure from K-means the dot product between feature vectors is used as similarity comparer in TextRank graphs and one-class SVM kernels:

$$Similarity(S_1, S_2) = v_{S_1}^T \cdot v_{S_2}$$

where v_{S_i} is the feature vector representation of sentence S_i. This allows for all three algorithms to use the same feature representaions and (near) equivalent measure of similarity.

Algorithm	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank	N, Adj	**0.4865**	0.4286	**0.4546**
One-class SVM	N, Adj	0.4089	0.3725	0.3891
K-means$_{centroid}$	N, Adj	0.4107	0.4169	0.4126
One-class SVM$_{inv}$	N, Adj	0.4563	0.4112	0.4315
K-means$_{TextRank}$	N, Adj	0.4473	**0.4377**	0.4410

(a) Keywords($n = 23$)

Algorithm	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank	N, Adj, AV, V	**0.4717**	0.4193	**0.4431**
One-class SVM	N, Adj, AV, V	0.4390	0.3981	0.4168
K-means$_{centroid}$	N, Adj, AV, V	0.3910	0.4037	0.3963
One-class SVM$_{inv}$	N, Adj, AV, V	0.4524	0.4064	0.4272
K-means$_{TextRank}$	N, Adj, AV, V	0.4288	**0.4221**	0.4240

(b) Filtered Word$_1$

Features	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank	N, Adj, AV, V	**0.4888**	**0.4346**	**0.4591**
One-class SVM	N, Adj, AV, V	0.4251	0.3868	0.4042
K-means$_{centroid}$	N, Adj, AV, V	0.3647	0.3724	0.3676
One-class SVM$_{inv}$	N, Adj, AV, V	0.4601	0.4131	0.4343
K-means$_{TextRank}$	N, Adj, AV, V	0.4257	0.4293	0.4264

(c) Filtered Word$_2$

Algorithm	**PoS-filter**	**Recall**	**Precision**	F_1
TextRank	N, Adj, AV, V	**0.4780**	**0.4254**	**0.4491**
One-class SVM	N, Adj, AV, V	0.4275	0.3928	0.4086
K-means$_{centroid}$	N, Adj, AV, V	0.4221	0.4224	0.4210
One-class SVM$_{inv}$	N, Adj, AV, V	0.4536	0.4079	0.4286
K-means$_{TextRank}$	N, Adj, AV, V	0.4190	0.4215	0.4189

(d) Filtered Word$_3$

Algorithm	**Recall**	**Precision**	F_1
TextRank	0.4522	0.4040	0.4259
One-class SVM	0.4106	0.3741	0.3907
K-means$_{centroid}$	0.4419	0.3965	0.4169
One-class SVM$_{inv}$	**0.4561**	**0.4100**	**0.4309**
K-means$_{TextRank}$	0.4537	0.4069	0.4275

(e) Named entities

Table 5.11: Equal sentence similarity

Each sub table in table 5.11 provides results for feature representations from the original experiments on the K-means algorithm (section 5.2). The TextRank and one-class SVM rows in the sub tables corresponds to the standard implementations with dot product between sentence feature vectors as similarity comparers. Two different versions of the K-means algorithm are tested here,

K-means$_{centroid}$ with centroid based and K-means$_{TextRank}$ with TextRank (dot product similarity) in cluster ranking. Both K-means variation uses K-means++ initial seeding. Apart from the standard one-class SVM, One-class SVM$_{inv}$, with inverted weights in the kernel matrix, is tested. To make the comparison as fair as possible, no position based boosting is involved in this test as the three algorithms cannot be boosted in precisely the same ways.

Table 5.11 show that TextRank outperforms the other two algorithms in this setting. When it comes to the performance of the one-class SVM versus K-means, they seem to give quite similar results. An interesting thing to notice is that the variations on the standard algorithms K-means$_{TextRank}$ and One-class SVM$_{inv}$ seem to outperform their standard algorithm counterparts (one-class SVM and K-means$_{centroid}$).

TextRank not only outperforms K-means and one-class SVM for this setting, the results displayed in table 5.11c is above any previously observed value for the unboosted version of the algorithm. This observation led to further testing of a position boosted version of TextRank with the dot product similarity measure. The results for this test was: $recall = 0.5099$, $precision = 0.4546$ and $F_1 = 0.4797$. These are the highest *recall* and F_1 scores observed for TextRank in this project.

Chapter 6

Comparison and discussion

In this section the results presented previously will be put in context of baselines and competitive systems. There are two different baselines used in this comparison. The first one is called Baseline_{Lead} and contains the leading sentences of a document using the standard sentence selection method described in section 3.1. This baseline is particularly strong in the domain of short news articles. Baseline_{Random} selects sentences at random from the document. This baseline is corpus independent and obviously needs to be outperformed by any system claiming to successfully perform the task of extractive summarization.

6.1 English

6.1.1 ROUGE evaluation

As mentioned in the section 2.3, the DUC2002 corpus and ROUGE toolkit have been used for the evaluation of other works on single document summarization. What follows is a comparison between the performance of the algorithms presented in this thesis and those from two papers very influential to this project[46, 30]. The ROUGE-scores of the different algorithms will, of course, also be compared to each other, as well as to the baselines. In extension to this, the theoretical limit to how well an extractive system can perform on the DUC2002 corpus will be discussed.

Algorithm	ROUGE N-gram(1,1)
TextRank	**0.4797**
K-means	0.4680
One-class SVM	0.4343
TextRank[46]	0.4708
K-means[30]	0.4791
Baseline_{Lead}	0.4649
Baseline_{Random}	0.3998

Table 6.1: Comparison table

Table 6.1 shows the different sentence ranking algorithms and their ROUGE scores. The first three rows corresponds to the approaches that have been developed in this project and their best ROUGE N-gram(1,1) F_1 scores. The fourth and fifth rows corresponds to scores reported in two other papers[30, 46], and the last two to the F_1 scores of the two baselines used.

The table shows that the results for the TextRank and K-means approaches are highly competitive for this corpus, as they beat both baselines, as well as closely compares to the results from the other papers. When it comes to the result for the one-class SVM algorithm it clearly beats Baseline_{Random}, while it falls short to the Baseline_{Lead}. The one-class SVM is the least domain specific implementation, since the position of sentences are not leveraged to the same extent as in the other implementations.

The original TextRank paper[46] was first presented in 2004, and does not specify whether the reported ROUGE score is the recall, precision or F_1 score. Since the ROUGE version used in DUC2004 only reported recall values[53], one could suspect that the reported score is in fact the recall. For this project, the highest observed recall score was 0.5099.

The K-means based system[30] presented results for both recall, precision and F_1 score. However, it did not specify any other ROUGE settings. For instance, enabling stemming can increase the ROUGE score. Since it was not specified, we assumed that the default settings were used, and all ROUGE evaluations were performed without stemming in this project. More over, their system was compared to other systems, for instance TextRank. The reported scores for TextRank were recall 0.4658, precision 0.4838 and F_1 0.4745, cited from the original TextRank paper[46]. But recall and precision are not reported in the original paper, and neither of these figures are consistent with the score 0.4708 which was the actual score reported in the original paper. Thereby no further conclusion could be drawn from the TextRank scores.

Extractive summarization systems are limited to selecting sentences from the document. They cannot produce or rewrite content and therefore cannot achieve a perfect F_1 score when compared to an abstractive gold summary. An experiment was devised to investigate how high the F_1 score could go for this corpus. The labeling algorithm proposed in appendix **B** was applied to select sentences that were similar to sentences in the abstractive summaries, which gave a F_1 score of 0.5657. There is no claim that this is the highest F_1 score that can be obtained, but it can be seen as a reasonable limit for what an extractive summarizer can achieve. The labeling algorithm "cheats" by looking at the gold standard abstracts, so algorithms based on sentence similarity seem unlikely to beat this.

6.1.2 Manual random inspection

In addition to the ROUGE evaluation, a manual inspection was also carried out to get a sense of the quality of the summaries. ROUGE only scores on coverage, and a good ROUGE score does not necessarily correspond to a good summary from a human point of view.

As stated in the introduction, a good summary should maintain the important information from the document. From the random inspection, it seems that most summaries are adequate in that regard. Below follows positive examples for the three algorithms.

TextRank

From the document AP900730-0116 (see C.1):

> East Germany's deposed Communist leader Erich Honecker is too sick to be held in jail but is fit enough to be tried, the official news agency ADN reported Monday. ADN,

> quoting a Health Ministry statement, said Honecker was still recovering from surgery for kidney cancer in January and was not well enough to be incarcerated. The medical panel concluded that seven other former Communist officials, who had been among Honecker's closest aides, were all fit to prosecute and to be placed in custody, ADN said. They had been arrested soon after Honecker's fall but were later released because of advanced age and poor health.

K-means

From the document FBIS3-1707 (see C.2):

> President Bill Clinton, trying to brush aside recent differences with London, today stressed Washington's special transatlantic relationship with Britain. Welcoming British Prime Minister John Major in Pittsburgh, where Major's grandfather and father once lived, Clinton said at the airport, "We're working together today to respond to the terrible tragedy in Bosnia to try to bring an end to the killing and to bring peace and to keep that conflict from spreading". Clinton will then share his Air Force One back to the nation's capital. Clinton and Major will meet again in June in Europe during the commemoration of the 50th anniversary of D-Day of the second world war. Major said Clinton would visit Britain, and perhaps the Oxford University, Clinton's alma mater, during the June visit.

One-class SVM

From the document AP900802-0180 (see C.3):

> THE INVASION: Iraq's troops, led by about 350 tanks, crossed the border at dawn Thursday, and seized the Kuwaiti palace and government buildings 40 miles away. CAUSE OF CONFLICT: President Saddam Hussein of Iraq accused Kuwait of stealing oil from its territory and forcing down oil prices through overproduction. Kuwait and U.N. diplomats dismissed Iraq's claim that it invaded at the request of Kuwaiti revolutionaries. The order also froze Kuwait property under U.S. jurisdiction – a move intended to keep Iraq from seizing it. The Soviets also condemned the invasion and stopped arms sales to Iraq.

However, the system is not perfect and will sometimes create summaries that are coherent but misses some important point in the original document. Here is an example produced by the one-class SVM (see C.4 for the full document):

> Supercomputers, satellites and the expertise of several hurricane forecasters predicted the destructive path Hurricane Hugo would follow, giving people plenty of time to flee the South Carolina coast. Forecasters at the National Hurricane Center used computer models to track Hugo's path into Charleston, S.C. Using the information from the satellite, supercomputers at the National Meteorological Center in Suitland, Md., send information to the hurricane center where a tracking model constantly changes to account for current weather conditions and the position of the hurricane. To determine the track of the storm, the forecasters analyze supercomputer predictions, satellite data, the history of similar storms and the current path of the hurricane.

While the summary seems coherent and easy to follow, it misses the point that the forecast models are unreliable and do not give as much information as the traditional forecast methods. This point was brought up by both human annotators.

Further more, the system can in some cases produce summaries that are hard to follow, where the information gap between the sentences is too great. Below is an example from the document LA062390-0001 (see C.5):

> Shortly after 2 p.m. Wednesday, San Diego resident Mohammad Nyakoui got a call from his wife, who is spending the summer in the Caspian Sea coastal city of Rasht, Iran, with her family . Hamid Biglari, a theoretical physicist at UC San Diego, was among those who organized a local group to gather contributions for the United Nations Fund for the Iran Earthquake . Donations for Iranian earthquake relief are being taken in San Diego by two groups: Mail checks to 3547 Camino del Rio South, Suite C, San Diego 92108, noting that donation is for Iranian Earthquake Relief Fund .

Not only is it difficult to understand the concept, the summary also fails to include the most important information: that there was an earthquake in Iran, and people worried about their relatives could not obtain any information within the first 48 hours.

6.2 Swedish

6.2.1 ROUGE evaluation

As no previous evaluation on this corpus was found, the results will only be compared to each other and the baselines.

Algorithm	ROUGE N-gram(1,1)
TextRank	**0.3593**
K-means	0.3539
One-class SVM	0.3399
Baseline_{Lead}	0.3350
Baseline_{Random}	0.3293

Table 6.2: Comparison table

Table 6.2 shows that all three approaches beat the baselines, and TextRank obtains the highest score for the Swedish corpus as well. An important thing to notice is that Baseline_{Lead} is much weaker here. The difference between Baseline_{Lead} and Baseline_{Random} is 0.0057 here, compared to 0.0651 in the DUC2002 corpus. The articles in the Wikipedia corpus are divided into several sub paragraphs. A stronger baseline would probably be to take the first sentence of each paragraph instead of taking the lead of the document. Basing boosting on this domain knowledge would probably also improve the results for both TextRank and k-means.

It should be noted that the reason for the lower ROUGE scores for this corpus is not because of the language per se, but rather that the domain changed into longer texts. Since the main evaluation was DUC2002, all improvements have been implemented with that corpus in mind. If the Wikipedia corpus would have been the corpus of the main evaluation, one could probably expect better performance.

The labeling algorithm proposed in appendix **B** was applied to the Wikipedia corpus as well and obtained a $F_1 - score$ of 0.4287. This can be seen as an indication that sentences from the articles are quite dissimilar to those in the abstracts.

6.2.2 Manual random inspection

The quality of the Wikipedia summaries are lower in general than the DUC2002 counterparts. While the general idea is often understood, the summaries might lack natural flow or fail to include important topics from the original text. Below are some examples of the summaries produced for the Swedish Wikipedia corpus.

One-class SVM

from a Wikipedia article on the Turkish language and its grammar (see C.6):

> Turkiskan ingår i den turkiska, eller västra, undergruppen av de sydvästturkiska (eller oghuziska) turkspråken. Turkiskans karakteristiska drag, som vokalharmoni, agglutinering och frånvaro av grammatiskt genus, är genomgående inom turkspråken och de altaiska språken. Den moderna turkiskan anses ofta ha skapats av Turkiets förste president, Kemal Atatürk, som införde det latinska alfabetet (och därmed också förbjöd användning av det arabiska) och lät genomföra omfattande språkreformer. Projekt som undersöker turkiska dialekter utförs av flera universitet, samt av en särskild arbetsgrupp i Turkiska språkföreningen, och för närvarande genomförs arbete med att samla och publicera forskningen som en uttömmande dialektatlas över turkiskan. Turkiskans vokaler är, i alfabetisk ordning, a, e, ı, i, o, ö, u, ü. Det förekommer inga diftonger i turkiskan och när två vokaler möts, vilket sker sällan och endast i lånord, behåller varje vokal sitt individuella ljud. Ett typiskt drag i turkiskan är vokalharmonin, vilket innebär att alla vokaler i ett ord måste överensstämma med varandra i uttalsposition. Turkiskan är ett agglutinerande och vokalharmoniserande språk som ofta använder affix, särskilt suffix (ändelser). Dessa lånord utgjorde omkring 20% av den dåvarande turkiskans ordförråd. I och med språkreformen på 1920-talet avskaffades det stora flertalet av de arabiska och persiska lånorden; de ersattes av dialektala, arkaiska och syntetiska ord, men även av synonymer, vilket gjorde turkiskan något ordfattigare. Det osmanska alfabetet angav endast tre olika vokaler-långa ā, ū och ī – och innehöll flera redundanta konsonanter såsom varianter av z (vilka var åtskilda i arabiskan men inte i turkiskan). Att korta vokaler saknades i det arabiska alfabetet gjorde det särskilt dåligt lämpat för turkiskan, som har åtta vokaler. Uppgiften att utarbeta det nya alfabetet och välja de nödvändiga modifieringarna för ljud som är särskilda för turkiskan gavs till en språkkommission som bestod av framträdande lingvister, akademiker och författare.

The extracted summary actually corresponds well to the gold standard abstract (also available in C.6) and manages to point out most of the important points. A direct contrast is presented below:

K-means

from a Wikipedia article on Thomas Aquinas (see C.7):

> De stora dragen och alla viktigare händelser i Thomas av Aquinos liv är kända, men biografier skiljer sig i vissa detaljer och datum från varandra. Denifles vän och elev, Dominic Prümmer, O.P. som var professor i teologi vid universitetet i Fribourg i Schweiz,

> upptog arbetet och publicerade Fontes Vitae S. Thomae Aquinatis, notis historicis et criticis illustrati. Den första faskikeln, Peter Calos biografi, utkom 1911 i Toulouse. Av den med Thomas samtide Bartolomeus av Lucca får vi veta att tidpunkten för Thomas födelse saknas vilket ledde till en osäkerhet om dennes exakta ålder . Paris höll fast honom; påvarna önskade honom nära sig; ordens Studia var angelägna om att åtnjuta hans undervisning; följaktligen finner vi honom i Anagni, Rom, Bologna, Orvieto, Viterbo, Perugia, i Paris igen, och slutligen i Neapel, alltjämt undervisande och skrivande, följande sin enda passion i livet att försvara de kristna doktrinerna. Thomas började omedelbart ombesörja förberedelserna för sin död.

The original text describes the life and works of Thomas Aquinas. The summary above, however, says practically nothing about these topics which makes it a particularly bad summary. The original text along with the gold standard abstract are included in C.7.

Chapter 7

Future work

During the work with this project several ideas for possible approaches and improvements emerged that were not, or at least not fully, implemented due to lack of time and/or resources. What follows is a description of some of those ideas.

7.1 Supervised learning

At the start of this project, one of the major goals was to compare unsupervised and supervised machine learning algorithms for sentence ranking. However, in the research phase it became clear that supervised approaches were dependent on the existence of good training sets and such sets seemed hard to come by. Two English corpora were found with sentences marked by humans as relevant or not[2, 14], but these seemed far from optimal. They both seemed a bit small for training purposes, the first one[2] contains 185 documents and the second[14] contains 183. Further more, the first corpus only used one annotator per document. As discussed in section 2.3.1, the problem with this is that different people find different sentences relevant. For this corpus, annotator comments were also available. After reading some of the comments, it seemed even more unfit for use:

> "May be unreliable due to difficulty/complexity of text topic!"
>
> "I am not very sure about this text."

A corpus using several annotators per document is available for Swedish texts[16], but the corpus is very small, it only contains 33 documents. Ideally one would want a large corpus with numerous annotators per document to train a supervised algorithm for sentence ranking.

Some work was put into using the sentence similarity comparers described in section 4.2, to mark sentences similar to those in the abstracts of the DUC2002 corpus as relevant or not. This method is described further in appendix B. However, as this automatic approach adds to the error rate of the system, the idea was abandoned.

7.2 Preprocessing

As mentioned in section 4.1, the text preprocessing steps is really what makes the system language dependent. The NLP libraries[11, 10] used in this project are dependent on language models

and in some cases no Swedish models were available. In other cases the models for Swedish performed worse than their English counterparts. New/improved models for Swedish would therefor be desirable.

Another interesting idea would be to use WordNet[47] instead of word stems for word comparisons. This would allow the system to view different synonyms as the same concept and thus, hopefully, be able to find similarities between sentences addressing the same topic with different words.

7.3 Spectral clustering

As mentioned in section 5.2, the K-Means clustering provided a quite uneven cluster distribution. Some clusters included many sentences, while others only included one. Which sentences ends up in which cluster is, of course, highly dependent on the feature representation and initial seeding. However, it would be interesting to explore other clustering algorithms to see if a more even distribution could be obtained. Some recent work[26] claims that spectral clustering can be used on a TextRank graph to divide it into sub graphs. These sub graphs are said to have roughly the same size and a high order of in cluster sentence similarity. As both TextRank graphs and clustering have been explored in this project, this would be an interesting approach to try.

7.4 Human evaluation

As mentioned in section 2.3.1, to determine if a summary is good or not is far form a trivial task. Different people have different opinions on what sentences from a document should be extracted to form a good summary. With this in mind, it would be interesting to construct a GUI (Graphical User Interface) where people could review extractive summaries. For example, a user could mark sentences in a summary as relevant or not. If such a system was constructed, the quality of summaries could be evaluated by majority vote.

As a presentation feature for this project, a simple GUI was made for viewing automatically created summaries. The GUI provides no review interaction with users, but could form as the basis for such a system.

7.5 Similarity comparer optimization

As stated in section 3.1, the time complexity of the algorithms are not the focus of this report. However, during evaluation on the Wikipedia corpus it became clear that the construction of the graphs and weight matrices were significantly slower for longer documents. The bottleneck here seemed to be the similarity comparison between sentences. It would be interesting to vectorize these comparisons as much as possible, to see if the running time decreases. That is to actually regard sentences as feature vectors in the comparers. If the sentences are represented as real number feature vectors, several of the comparers could probably make use of fast linear algebra libraries.

Further more, vector representations of similarity comparers would allow for more equivalent testing between the three algorithms. A small experiment was conducted for 3 feature representations in section 5.4. A vectorized approach would not only allow for equal feature representation. The vectorized similarity measures could also be used in the cluster assignment step in k-means, which means that all three algorithms could use the same similarity measure.

The vectorization of the comparers is only one possible optimization strategy, there may be several others.

Chapter 8

Conclusion

This thesis has described the implementation and evaluation of a system for automatic, extractive single document summarization. As the basis for this system three algorithms for sentence relevance ranking were explored. The first one was the well established TextRank, the second was based on K-means clustering and the third on one-class SVM. All three algorithms are considered unsupervised machine learning algorithms and therefore demands no language specific training sets. The only language dependence of the system is that needed for text preprocessing.

The system has been evaluated mainly using a corpus consisting of short news articles in English. The main evaluation method used was the well known ROUGE toolkit. Using this evaluation tool, TextRank was found to yield the best result on the corpus. As an attempt to improve the results, different variations of TextRank not mentioned in the original paper on the algorithm were tested. The variations did not improve the result significantly. It was instead found that domain specific boosting could make a bigger difference.

When it came to the other algorithms, the k-means sentence ranking yielded a ROUGE-score comparable to that of TextRank while one-class SVM approach fell a bit short. Both algorithms managed to outperform the baseline of selecting sentences at random from a document, but the one-class SVM performs worse then selecting the lead sentences from a text to form a summary.

Even though the main focus of this project was summarization of documents in English, the system was also tested for Swedish. The domain here were longer, Wikipedia articles. Evaluation results outperforming the two baselines described above were observed also in this domain. Although the baseline of using the head of a document as a summary is not as strong in this domain, these results can be seen as a strong indication of the relative language independence of the system.

As expected the system does not produce as perfect summaries of those written by humans in an abstract fashion. To achieve perfect coherent summaries one would probably have to switch paradigm to abstractive summarization. However, in our opinion, this system can be used to produce adequate summaries in the domain it was mainly evaluated in: short news articles in English. To perform well on other domains, the base algorithms could probably be used in combination with domain specific knowledge/boosting.

Bibliography

[1] http://berouge.com/default.aspx, May 2012.

[2] http://clg.wlv.ac.uk/projects/CAST/corpus/index.php, May 2012.

[3] http://commons.apache.org/math/, May 2012.

[4] http://duc.nist.gov/, May 2012.

[5] http://en.wikipedia.org/wiki/Cosine_similarity, May 2012.

[6] http://en.wikipedia.org/wiki/Laplacian_matrix, May 2012.

[7] http://en.wikipedia.org/wiki/Search_engine_indexing, May 2012.

[8] http://en.wikipedia.org/wiki/Wikipedia:Featured_article_criteria, May 2012.

[9] http://kavita-ganesan.com/rouge-howto, May 2012.

[10] http://nlp.stanford.edu/software/index.shtml, May 2012.

[11] http://opennlp.apache.org/, May 2012.

[12] http://search.cpan.org/~kubina/Text-Corpus-Summaries-Wikipedia-0.21/lib/Text/Corpus/Summaries/Wikipedia.pm, May 2012.

[13] http://www-nlpir.nist.gov/projects/duc/past_duc/duc2002/test.html, May 2012.

[14] http://www-nlpir.nist.gov/related_projects/tipster_summac/cmp_lg.html, May 2012.

[15] http://www.maltparser.org/, May 2012.

[16] http://www.nada.kth.se/iplab/hlt/kthxc/showsumstats.php, May 2012.

[17] http://www.nist.gov/tac/, May 2012.

[18] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. Np-hardness of euclidean sum-of-squares clustering. *Mach. Learn.*, 75(2):245–248, May 2009.

[19] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the twenty-second annual symposium on Computational geometry*, SCG '06, pages 144–153, New York, NY, USA, 2006. ACM.

[20] David Arthur and Sergei Vassilvitskii. k-means++: the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '07, pages 1027–1035, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[21] Araly Barrera and Rakesh Verma. Automated extractive single-document summarization: beating the baselines with a new approach. In *Proceedings of the 2011 ACM Symposium on Applied Computing*, SAC '11, pages 268–269, New York, NY, USA, 2011. ACM.

[22] Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:2002, 2002.

[23] Asa Ben-Hur, David Horn, Hava T. Siegelmann, and Vladimir Vapnik. Support vector clustering. *JOURNAL OF MACHINE LEARNING RESEARCH*, 2:125–137, 2001.

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

[25] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):27:1–27:27, May 2011. Software available at http://www.csie.ntu.edu.tw/c̃jlin/libsvm.

[26] S. Cotter. Improving search results with automated summarization and sentence clustering. Master's thesis, McAnulty College and Graduate School of Liberal Arts, Department of Computational Mathematics, 2012.

[27] Robert L. Donaway, Kevin W. Drummey, and Laura A. Mather. A comparison of rankings produced by summarization evaluation measures. In *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*, NAACL-ANLP-AutoSum '00, pages 69–78, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[28] H. P. Edmundson. New methods in automatic extracting. *J. ACM*, 16(2):264–285, April 1969.

[29] Günes Erkan and Dragomir R. Radev. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479, December 2004.

[30] René Arnulfo García-Hernández, Romyna Montiel, Yulia Ledeneva, Eréndira Rendón, Alexander Gelbukh, and Rafael Cruz. Text Summarization by Sentence Extraction Using Unsupervised Learning. In *Proceedings of the 7th Mexican International Conference on Artificial Intelligence: Advances in Artificial Intelligence*, MICAI '08, pages 133–143, Berlin, Heidelberg, 2008. Springer-Verlag.

[31] Martin Hassel. *Resource Lean and Portable Automatic Text Summarization*. PhD thesis, School of Computer Science and Communication, Royal Institute of Technology, Stockholm, Sweden, June 2007.

[32] Tsutomu Hirao, Hideki Isozaki, Eisaku Maeda, and Yuji Matsumoto. Extracting important sentences with support vector machines. In *Proceedings of the 19th international conference on Computational linguistics*, volume 1 of *COLING '02*, pages 1–7, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics.

[33] V. Jethava, A. Martinsson, C. Bhattacharyya, and D. Dubhashi. Lovász theta function, SVMs and finding dense subgraphs. Preprinted version, Chalmers University of Technology, 2012.

[34] Martin C Johansson and Pontus M Lindström. Keyword Extraction using Machine Learning. Master's thesis, Chalmers tekniska högskola, 2010.

[35] Rohit J. Kate. A dependency-based word subsequence kernel. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 400–409, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[36] J. Kleinberg and É. Tardos. *Algorithm Design*. Pearson international edition. Pearson/Addison-Wesley, 2006.

[37] C. Kruengkrai and C. Jaruskulchai. Using oneclass svms for relevant sentence extraction. In *Proceedings of the 3rd International Symposium on Communications and Information Technologies*, Songkhla, Thailand, 2003.

[38] Daniel S. Leite, Lucia H. M. Rino, Thiago A. S. Pardo, and Maria das Graças V. Nunes. Extractive automatic summarization: Does more linguistic knowledge make a difference? In *Proceedings of the Second Workshop on TextGraphs: Graph-Based Algorithms for Natural Language Processing*, pages 17–24, Rochester, NY, USA, 2007. Association for Computational Linguistics.

[39] Daniel Saraiva Leite and Lucia Helena Rino. Combining multiple features for automatic text summarization through machine learning. In *Proceedings of the 8th international conference on Computational Processing of the Portuguese Language*, PROPOR '08, pages 122–132, Berlin, Heidelberg, 2008. Springer-Verlag.

[40] Chin Y. Lin. Rouge: A Package for Automatic Evaluation of Summaries. In Marie F. Moens and Stan Szpakowicz, editors, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, Barcelona, Spain, 2004. Association for Computational Linguistics.

[41] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using N-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, volume 1 of *NAACL '03*, pages 71–78, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

[42] Stuart P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, IT-28(2):129–137, March 1982.

[43] H. P. Luhn. The automatic creation of literature abstracts. *IBM J. Res. Dev.*, 2(2):159–165, April 1958.

[44] Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. In *Proceedings of the 3rd International Workshop on Algorithms and Computation*, WALCOM '09, pages 274–285, Berlin, Heidelberg, 2009. Springer-Verlag.

[45] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schtze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[46] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. In *Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, 2004.

[47] George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. Software available at http://wordnet.princeton.edu/.

[48] Ani Nenkova and Kathleen McKeown. Automatic Summarization. *Foundations and Trends in Information Retrieval*, 5(2–3):103–233, 2011.

[49] G. J. Rath, A. Resnick, and T. R. Savage. The formation of abstracts by the selection of sentences. Part I. Sentence selection by men and machines. *American Documentation*, 12:139–141, 1961.

[50] G. J. Rath, A. Resnick, and T. R. Savage. The formation of abstracts by the selection of sentences. Part II. The reliability of people in selecting sentences. *American Documentation*, 12:141–143, 1961.

[51] K. Sarkar, M. Nasipuri, and S. Ghose. Using machine learning for medical document summarization. *International Journal of Database Theory and Application*, 4:31–48, 2011.

[52] Bernhard Schölkopf, John C. Platt, John C. Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Comput.*, 13(7):1443–1471, July 2001.

[53] Jonas Sjöbergh. Older versions of the rougeeval summarization evaluation system were easier to fool. In *Information Processing & Management*, pages 1500–1505, 2007.

[54] Karen Spärck Jones. Automatic summarising: The state of the art. *Information Processing and Management*, 43(6):1449–1481, November 2007.

[55] Mehdi Yousfi-Monod, Atefeh Farzindar, and Guy Lapalme. Supervised machine learning for summarizing legal documents. In *Proceedings of the 23rd Canadian conference on Advances in Artificial Intelligence*, AI'10, pages 51–62, Berlin, Heidelberg, 2010. Springer-Verlag.

Appendix A

File formats and running the ROUGE script

Note that this part is more of a practical description on how the summarization system is evaluated using ROUGE, meant for anyone who wants to replicate the results.

The first step is to prepare the files that will used in the evaluation process. A few different input formats are accepted, where SEE, short for *Summary Evaluation Environment*, seems to be the most straight forward one (by inspection of the readme file and supplied examples)[9]. The system summary of each document is saved as an HTML document containing the extracted sentences, and the golden standard summaries are saved as separate HTML documents in the same format. Then *settings.xml* is generated, which pairs the system summaries with golden summaries.

In the evaluations that were conducted, the summarization system was applied to all documents of the DUC2002 corpus and the summaries were saved using the document ID, e.g. *genAP900427-0083.html* where *gen* is short for *generated*. The golden standards were saved in a similar manner, e.g. *gold{X}AP900427-0083.html*, where *{X}* is replaced with a character that represents the human annotator. An example of the contents of these files can be seen in fig. A.1. The file *settings.xml* can then be generated, an example is shown in fig. A.2.

```
<html>
  <head>
    <title>genAP900427-0083</title>
  </head>
  <body bgcolor="white">
    <a name="0">[0]</a> <a href="#0" id="0">The lens cover...</a>
    <a name="1">[1]</a> <a href="#1" id="1">...</a>
    ...
  </body>
</html>
```

Figure A.1: SEE format example

```
<ROUGE_EVAL version="1.55">
  <EVAL ID="0">
    <PEER-ROOT>data/Summaries</PEER-ROOT>
    <MODEL-ROOT>data/Summaries</MODEL-ROOT>
    <INPUT-FORMAT TYPE="SEE"></INPUT-FORMAT>
    <PEERS>
      <P ID="C">genAP900427-0083.html</P>
    </PEERS>
    <MODELS>
      <M ID="0">goldHAP900427-0083.html</M>
      <M ID="1">goldJAP900427-0083.html</M>
    </MODELS>
  </EVAL>
  ...
</ROUGE_EVAL>
```

Figure A.2: An example of *settings.xml*

The ROUGE evaluation system is invoked by running a Perl script from the shell. A number of different arguments may be passed to the script, but only the ones relevant to the evaluation in this report are included below:

```
./ROUGE-1.5.5.pl -e data -a -n 1 -x data/settings.xml
```

The arguments `-e data` and `data/settings.xml` simply tells the script which directory that should be the working directory in the evaluation and where to find the settings file from the current working directory. The others are more interesting:

-a to run evaluation on all systems (there will only be 1 system but there may be 1 or more golden standards). In case of more than 1 golden standard, the system will be compared to all of them, and the highest score that was found is reported.

-n 1 to set $N = 1$ for the N-grams, i.e. only unigrams (single terms) are considered in this evaluation.

-x to explicitly disable ROUGE-L calculation which is not desired here.

An example of the ROUGE script output can be seen in fig. A.3. To collect the comparison data in the result parts of this report, the system was configured with some set of options and then applied to the documents of the DUC2002 corpus. The resulting summaries and golden standards were written to files as described above and the ROUGE script was run to collect the F_1 score.

```
C ROUGE-1 Average_R: 0.45736 (95%-conf.int.  0.45003 - 0.46440)
C ROUGE-1 Average_P: 0.44824 (95%-conf.int.  0.44151 - 0.45564)
C ROUGE-1 Average_F: 0.45146 (95%-conf.int.  0.44448 - 0.45826)
```

Figure A.3: ROUGE script output

Appendix B

Automatic sentence labeling

Since no suitable data set containing extractive summaries was found in the literary survey for this project, an idea of automatic labeling of sentences was formed.

B.1 Labeled extractor

The idea was to use the sentence similarity comparers developed for TextRank to find sentences in a document similar to those in the golden abstracts. The sentences most similar to those in the gold standard abstracts would then be labeled as relevant, while the others would be labeled irrelevant.

For each sentence in a document, the sentence is compared to all sentences in the golden abstracts. The sentences that were the most similar to the abstract sentences, were then selected to form an extractive gold standard summary.

Using this label data was abandoned since the risk of mislabeling seemed too great. However, the labeled extractor was used to analyze the limitation of extractive summarization on the evaluation corpora.

Appendix C

Example documents

C.1 DUC2002: AP900730-0116

Document contents

Panel: Honecker Unfit to be Held but Fit to be Tried

East Germany's deposed Communist leader Erich Honecker is too sick to be held in jail but is fit enough to be tried, the official news agency ADN reported Monday. ADN, quoting a Health Ministry statement, said Honecker was still recovering from surgery for kidney cancer in January and was not well enough to be incarcerated. But a medical exam by a team of doctors found him in condition to be questioned and to stand trial, ADN said. Honecker, 77, ruled East Germany for 19 years until he was ousted in October as a wave of pro-democracy demonstrations swept the country, leading to the peaceful overthrow of the Communist government and the opening of the Berlin Wall. The medical panel concluded that seven other former Communist officials, who had been among Honecker's closest aides, were all fit to prosecute and to be placed in custody, ADN said. It said former secret police chief Erich Mielke, former economics czar Guenter Mittag and former labor chief Harry Tisch were under arrest, while the other four remained free. All three had been members of the ruling Politburo under Honecker. Mielke was arrested on Thursday. Honecker and the seven are charged with corruption and abuse of office. They had been arrested soon after Honecker's fall but were later released because of advanced age and poor health. Honecker has been staying at a Soviet military hospital outside East Berlin. ADN quoted Federal Prosecutor Guenter Seidel as saying he intended to continue his investigation against the former East German leaders, although their poor health often disrupted the proceedings.

Annotator D

East Germany's deposed Communist leader Erich Honecker is too sick to be held in jail, but is fit enough to be tried. Honecker is still recovering from kidney cancer surgery and is not well enough to be incarcerated. Honecker 77, ruled East Germany for 19 years until he was ousted in October 1989 as a wave of pro-democracy demonstrations led to a peaceful overthrow of the Communist government. Honecker and seven others are charged with corruption and abuse of office. The federal prosecutor said he plans to

continue the investigation against the former East German leaders, despite their poor health.

Annotator G

East Germany's deposed Communist leader, Erich Honecker, has been declared too sick to be in jail but fit enough to be tried. He is still recovering from kidney cancer surgery in January. A medical panel concluded that seven other former Communist officials–all close aides to Honecker–were fit to prosecute and to be placed in custody despite advanced age and poor health. Along with Honecker, they are charged with corruption and abuse of office. Honecker has been staying at a Soviet military hospital outside East Berlin. Although their poor health often disrupted proceedings, investigations against the former East German leaders will continue.

C.2 DUC2002: FBIS3-1707

Document contents

President Clinton, John Major Emphasize 'Special Relationship'

Language: English Article Type:BFN [Text] Washington, February 28 (XINHUA) – U.S. President Bill Clinton, trying to brush aside recent differences with London, today stressed Washington's special transatlantic relationship with Britain. Welcoming British Prime Minister John Major in Pittsburgh, where major's grandfather and father once lived, Clinton said at the airport, "We're working together today to respond to the terrible tragedy in Bosnia to try to bring an end to the killing and to bring peace and to keep that conflict from spreading". For his part, Major said, pressure would be increased for the peace that every sensitive person wishes to see in that war-torn and troubled land. On Russia, Major said "A Russia that's a good neighbor to the United States and West would be one of the finest things that this generation could hand down to the next". Clinton will then share his Air Force One back to the nation's capital. Major will spend a night at the White House, the first foreign head of state to have this honor since Clinton became President. On Tuesday [1 March], the two leaders will begin their discussions on a wide range of issues including Russia, Bosnia, Northern Ireland and the world trade. The two will also discuss Northern Ireland and "what to do with NATO," Clinton said. Clinton and major will meet again in June in Europe during the commemoration of the 50th anniversary of D-Day of the second world war. Major said Clinton would visit Britain, and perhaps the Oxford University, Clinton's alma mater, during the June visit.

Annotator E

In trying to brush aside recent differences with London, U.S. President Bill Clinton stressed Washington's special relationship with Britain. These remarks came as Clinton welcomed Prime Minister John Major in Pittsburgh where Major's grandfather and father once lived. Following the visit to Pittsburgh, Clinton and Major will return to the White House, where Major will spend the night, the first foreign head of state to do so since Clinton became president. High on the agenda for talks between the two men are the tragedy in Bosnia as well as Russia, Northern Ireland, and world trade. Clinton and Major will meet in June again during D-Day commemoration ceremonies.

Annotator J

In a welcome to British Prime Minister John Major today at the Pittsburgh airport, President Clinton said the two nations were working together to end the conflict in Bosnia. Major, after visiting Pittsburgh where his father and grandfather once lived, will travel with the President back to Washington on Air Force One. On Tuesday, the two leaders will begin discussing a wide range of issues including Russia, Bosnia, Northern Ireland, and world trade. Clinton and Major will meet again in June during D-Day 50th anniversary events in Europe. Major said Clinton would visit Britain, including possibly Oxford University, Clinton's alma mater.

C.3 DUC2002: AP900802-0180

Document contents

Basic Facts of Iraq's Invasion of Kuwait

THE INVASION: Iraq's troops, led by about 350 tanks, crossed the border at dawn Thursday, and seized the Kuwaiti palace and government buildings 40 miles away. Early Friday, the invaders controlled Kuwait city, the capital. The soldiers were also making sweeps through the southern oilfields, according to residents and Lloyds' insurance service in London. Diplomatic sources estimate more than 200 Kuwaitis were killed or wounded, mainly from the Emiri Guard which bore the main brunt of the invasion. Kuwait's Sheik Saad al-Abdullah al-Sabah fled to safety in Saudi Arabia. THE FORCES: Iraq, a country of more than 17 million, has the Arab world's most battle-trained army, and had massed more than 100,000 soldiers on the Kuwaiti border. Kuwait, an oil-rich city-state, has 1.9 million residents – 60 percent foreigners – and an army of 20,300 soldiers. CAUSE OF CONFLICT: President Saddam Hussein of Iraq accused Kuwait of stealing oil from its territory and forcing down oil prices through overproduction. Kuwait and U.N. diplomats dismissed Iraq's claim that it invaded at the request of Kuwaiti revolutionaries. REACTION: The U.N. Security Council voted 14-0 to condemn the invasion. President Bush denounced it as "naked agression". He froze Iraq's assets in the United States and blocked almost all Iraqi imports, including oil. The order also froze Kuwait property under U.S. jurisdiction – a move intended to keep Iraq from seizing it. The Soviets also condemned the invasion and stopped arms sales to Iraq. Kuwait's U.S. ambassador said the nation had asked for American military intervention. A Pentagon source said a U.S. naval group was diverted toward the Persian Gulf. Oil prices soared in frenzied trading amid fears the invasion would reduce the supply of oil.

Annotator H

100,000 Iraqi soldiers crossed into Kuwait Thursday, led by 350 tanks. By Friday Iraq controlled the capital. Soldiers swept through southern oilfields. Over 200 Kuwaitis were killed or wounded, mainly the Emiri Guard. Sheik al-Sabah fled to Saudi Arabia. Iraqi President Saddam Hussain said Kuwait was stealing oil and forcing down prices through overproduction and Kuwaiti revolutionaries requested that Iraq invade, which Kuwait denies. The UN condemned the invasion. The US froze Iraqi and Kuwaiti assets in the US and blocked Iraqi imports. The Soviets stopped arms sales to Iraq. Kuwait

asked for US military intervention. A US naval group was diverted to the Persian Gulf. Oil prices soared.

Annotator I

Iraqi troops invaded Kuwait and by early Friday controlled Kuwait city, the capital. Soldiers also were sweeping through the oilfields. More than 200 Kuwaiti casualties are reported. The Sheik fled to Saudi Arabia. Iraq, with 17 million people, amassed 100,000 of the most battle-trained army in the Arab world on the border. Kuwait has 1.9 million residents, 60 percent foreigners, and an army of 20,300. Iraq accused Kuwait of stealing and overproducing oil. The U.N. Security Council, the Soviets, and President Bush have condemned the attack. A U.S. naval group is moving toward the area, as oil prices soar.

C.4 DUC2002: AP890922-0167

Document contents

Forecasting Aided By Supercomputers, But Still An Uncertain Science

Supercomputers, satellites and the expertise of several hurricane forecasters predicted the destructive path Hurricane Hugo would follow, giving people plenty of time to flee the South Carolina coast. But hurricane tracking remains an uncertain science. Forecasters at the National Hurricane Center used computer models to track Hugo's path into Charleston, S.C. "All the world's knowledge about meteorological conditions and forecasting changes in those conditions is embodied in those models," said Thomas Pyke, head of National Oceanic and Atmospheric Administration's satellite service. Pinpointing the exact point of Hugo's landfall was difficult, but forecasters said Friday that the landfall was predicted in time for evacuation. "Overall, I think the tracking models gave us a very good idea where Hugo would be so officials in South Carolina could act in a timely manner," said research meteorologist Colin McAdie. The real forecasting problem with Hugo was predicting the intensity of the storm, which was upgraded to a Category 4 hurricane just hours before it slammed into Charleston. "It is very difficult to predict changes in intensity because we don't have very reliable computer models for that," McAdie said. "We really need to improve on our forecasting ability of strength". The hurricane specialists were surprised by the last-minute increase in wind speed, which was reported to them by Air Force reconnaissance. Hurricane specialist Gil Clark, who has tracked hurricanes for 35 years, said that a couple of decades ago, the only forecasting tools were reports from aircraft or ships. "We had no radar or satellites then, so needless to say our forecasts were less accurate," Clark said. In the late 1960s, the weather service began using satellites to obtain a global weather picture. Information from the satellite is used to improve the accuracy of the large-scale models that television viewers see every night. Using the information from the satellite, supercomputers at the National Meteorological Center in Suitland, Md., send information to the hurricane center where a tracking model constantly changes to account for current weather conditions and the position of the hurricane. To determine the track of the storm, the forecasters analyze supercomputer predictions, satellite data, the history of similar storms and the current path of the hurricane. Then they make an educated guess about the landfall. Meteorology professor Kerry A. Emmanuel of the Massachussetts Institute of Technology criticizes the current forecasting system.

"Congress and the American people are suffering from the collective delusion that our data problems have been solved by satellites and that just isn't true," Emmanuel said. Satellites can give a "pretty picture," he said, but not enough information about the wind and temperatures that affect a hurricane's path. "Most of the information actually used to predict hurricanes comes from flying airplanes into the hurricane, and they do a very good job," Emmanuel said. Forecasters say the accuracy of satellite pictures is improving every year so long-range forecasting should become more precise. "We have to remember that those models used are only guidance products," Pyke said, "and that it's ultimately the job of the forecaster to predict the storm's path."

Annotator A

Supercomputers and satellites helped predict the path and landfall of Hurricane Hugo in time to allow evacuation of Charleston, S.C., but hurricane tracking is not an exact science. The intensity of Hugo was upgraded on the basis of Air Force reconnaissance just hours before it hit Charleston. There are no reliable computer models for forecasting strength. To predict the path, supercomputers use satellite data to make constantly changing predictions. Forecasters analyze the predictions, satellite data, the history of similar storms and the current path to make an educated guess about the landfall. The best information for predicting hurricanes still comes from flying airplanes into the hurricane.

Annotator H

Hurricane tracking remains an uncertain science, but forecasters' predictions of Hugo's destructive path gave people time to flee South Carolina's coast. Their expertise was assisted by satellite information sent by supercomputers at the National Meteorological Center in Suitland, Maryland. At the National Hurricane Center, a tracking model embodying knowledge of meteorological conditions and indicators of change constantly reflected weather conditions and storm position. Pinpointing landfall was difficult. Also, computer models can't reliably predict a storm's intensity. Air Force reconnaissance planes flying into the storm still provide important data about wind speed and temperatures. Before the satellites of the late 1960s, reports from aircraft or ships were the only forecasting tools.

C.5 DUC2002: LA062390-0001

Document contents

SAN DIEGANS AWAIT WORD ON IRAN RELATIVES

Shortly after 2 p.m. Wednesday, San Diego resident Mohammad Nyakoui got a call from his wife, who is spending the summer in the Caspian Sea coastal city of Rasht, Iran, with her family . "I asked her what she was doing, and she said she was watching the World Cup," Nyakoui recalls . About 15 minutes after she hung up, Rasht and other cities in Iran's northern region were rocked by an earthquake that killed thousands . Back home in San Diego, it was several hours before Nyakoui even heard news of the quake. When he did, some quick time-zone arithmetic showed that he had talked to his wife just minutes before the quake happened . "I was very worried," Nyakoui said.

"I tried through the Red Cross to call Iran, but they told us they couldn't do anything before 48 hours passed" . One of about 25,000 San Diego Iranians in the same anxious state, Nyakoui was luckier than most: His wife managed to call him a second time that day, about eight hours after the quake, to tell him she and their two children were safe . Phone lines remained jammed Friday, other Iranian-Americans in San Diego said . "There's a sense of panic. It's just impossible to get through," said Houshang Ghashghai, who teaches political science at San Diego State University . Already though, some of the worried immigrants were turning their fear into action by mobilizing relief efforts for Iran . Hamid Biglari, a theoretical physicist at UC San Diego, was among those who organized a local group to gather contributions for the United Nations Fund for the Iran Earthquake . The group has set up two offices and phone lines. They are collecting medicine, food and a variety of living supplies, including blankets, light clothing and flashlights. The supplies will be transported by the United Nations, Biglari said . Another group gathering medical supplies is Southwest Medical Teams, an organization whose last major international effort was sending volunteers and medical supplies to Armenia after the 1988 quake there . Southwest Medical Teams won't be sending any volunteers this time, said director Barry La Forgia . "That's at the advice of the State Department. They couldn't ensure our safety over there, and we don't want to endanger any of our volunteers," La Forgia said . WHERE TO SEND AID Donations for Iranian earthquake relief are being taken in San Diego by two groups: United Nations Fund for the Iran Earthquake (467-1120 or 456-4000) – Collecting antibiotic and analgesic drugs, collapsible plastic water containers, plastic sheeting, 220-volt portable generators of less than 10 kilowatts, tents, blankets, dry food, light clothing, flashlights and lighting equipment. Bring items or mail checks to 4540 Kearny Villa Road, Suite 214, San Diego 92123; or to 7509 Girard Ave., Suite A, La Jolla 92037. Checks should be made out to the U.N. Fund for the Iran Earthquake . Southwest Medical Teams (284-7979) – Through Thursday, collecting sutures, surgical gloves, antibiotics, analgesics (including aspirin), vitamins, ophthalmic solutions and empty blood collection bags. Shipment will be sent July 2. Cash donations will be used to pay to fly the supplies to Iran. Mail checks to 3547 Camino del Rio South, Suite C, San Diego 92108, noting that donation is for Iranian Earthquake Relief Fund . Other agencies accepting donations for Iranian earthquake victims: Adventist Development and Relief Agency 12501 Old Columbia Pike Silver Spring, Md. 20904 (301) 680-6380 American Red Cross Iran Earthquake Disaster P.O. Box 37243 Washington, D.C. 20013 (800) 842-2200 Bank Melli Iran Iran Quake Relief Assistance Account No. 5000 628 Madison Ave . New York, N.Y. 10022 U.S. Committee for UNICEF 333 E. 38th St . New York, N.Y. 10016 (212) 686-5522

Annotator A

The 25,000-strong Iranian community in San Diego has responded quickly to the earthquake that hit northern Iran on Wednesday, California time. Many have turned anxiety over the fate of relatives and friends into action by mobilizing relief efforts, setting up two offices (467-1120 and 456-4000) to gather contributions for the United Nations Fund for the Iran Earthquake. The Iranian group is collecting medicine, light clothing, blankets, and flashlights among other things. Another group collecting medical supplies is Southwest Medical Teams (284-7979). Donations for Iranian earthquake relief may also be sent to the Red Cross, UNICEF, Bank Meli Iran, and the Adventist Development and Relief Agency.

Annotator E

San Diego's Iranian community was in a anxious state after receiving word of the earthquake in Iran. Most of the 25,000 Iranians living in San Diego were told they would have to to wait at least 48 hours for news of family and friends. Mohammad Nyakoui was more fortunate. He had talked with his wife just before the earthquake and she was able to call him eight hours after the earthquake to let him know that she and their two children were alright. The Iranian Community has begun to organize to receive donations of money as well as food, clothing, and medicine.

C.6 Wikipedia: 9edc252b1dbb27636e2a9ecbb96e14d3

Document contents

Turkiskan ingår i den turkiska, eller västra, undergruppen av de sydvästturkiska (eller oghuziska) turkspråken. Närliggande språk är gagauziska (talat främst i Gagauzien i Moldavien), balkangagauziska och khorasanturkiska, azerbajdzjanska samt, på längre håll, turkmeniska. Turkspråken är en språkfamilj som omfattar ungefär 30 levande språk som talas i Östeuropa, Centralasien och Sibirien. De ingår enligt en del forskare i en större familj, altaiska språk, som även inkluderar mongolspråk. Under främst 1800-talet ville en grupp lingvister inordna turkspråken och mongolspråken i den uralaltaiska storfamiljen, där finska, estniska och ungerska ingår i Europa liksom en stor mängd mindre språk i europeiska och asiatiska Ryssland. Omkring 40 procent av alla som talar turkspråk talar turkiska. Turkiskans karakteristiska drag, som vokalharmoni, agglutinering och frånvaro av grammatiskt genus, är genomgående inom turkspråken och de altaiska språken. Det finns en hög grad av ömsesidig förståelse mellan turkisktalande och talare av de andra turkspråken, inklusive azeri, turkmeniska, qashqai och gagauziska. Däremot kan inte talare av övriga turkspråk utan undervisning och träning läsa turkiska då detta språk har övergått till latinska bokstäver. Turkiskan går tillbaka till fornturkiskan. De tidigaste kända turkspråkiga inskrifterna återfinns i nuvarande Mongoliet, som Bugut-inskrifterna med sogdiska alfabetet under det första gökturkiska khanatet, vilka dateras till andra hälften av 500-talet. De två monumentala Orchoninskrifterna, som restes till fursten Kul Tigins och hans bror kejsaren Bilge Khans ära och härstammar från något tillfälle mellan 732 och 735, utgör en annan betydelsefull tidig lämning. Efter att dessa monument upptäckts och utgrävts av ryska arkeologer i området kring Orchondalen vid floden Orchon i nuvarande Mongoliet från 1889 till 1893 fastslogs att språket i inskrifterna var fornturkiska skrivet med orchonskrift, som också har kallats "turkiska runor" på grund av yttre likhet med de germanska runorna. Fornturkiskan kom att delas i flera grenar, däribland den västligt oghuziska. Under tidig medeltid (cirka 500-tal till 1000-tal) spreds folkgrupper som talade turkspråk över Centralasien så att de täckte ett vidsträckt geografiskt område från Sibirien till Europa och Medelhavet. Seldjukerna i synnerhet spred sitt oghuziska språk, den direkta föregångaren till dagens turkiska, till Anatolien från 1071, då de fick herravälde över den östra delen av halvön efter slaget vid Manzikert. I och med islams spridning på 1000-talet började turkspråk skrivas med det arabiska alfabetet (karachanidiskan). Under 1000-talet publicerade också en tidig språkforskare, Kaşgarlı Mahmud från Karakhanidkhanatet, den första större ordboken över turkspråk, Divânü Lügati't-Türk, vilken också innehöll den första kända kartan över turkspråkstalares geografiska utbredning. Seldjukerna spred under 1200-talet turkiskan till Anatolien. Där inleddes på 1200-talet fornanatoliskturkiskan, ett förstadium till osmanskan (den

turkiska som kom att utvecklas i det osmanska riket). Den osmanska turkiskan delas in i tre perioder: fornosmanskan (fram till 1400-talet), medelosmanskan (1500- och 1600-talen) samt nyosmanskan (1700- och 1800-talen). Osmanskan hade en stor mängd arabiska och persiska lånord (dessa utgjorde som mest omkring 20 procent av ordförrådet).[källa behövs] Efter att karachaniderna och seldjukerna, som betraktas som de kulturella föregångarna till osmanerna, antagit islam omkring år 950, upptog dessa staters administrativa språk en ganska stor samling lånord från arabiska och persiska. Turkisk litteratur från den osmanska perioden, i synnerhet osmansk divanpoesi, var starkt influerad av persiska. Bland annat upptogs persiska versmått och en stor mängd lånord. Under de över 600 år då det osmanska riket existerade (cirka 1299–1922) var dess litterära och officiella språk en blandning av turkiska, persiska och arabiska, vilken skilde sig avsevärt från den vardagliga talade turkiskan, och betecknas osmanska. Den moderna turkiskan anses ofta ha skapats av Turkiets förste president, Kemal Atatürk, som införde det latinska alfabetet (och därmed också förbjöd användning av det arabiska) och lät genomföra omfattande språkreformer. De arabiska och persiska lånorden ersattes i stor utsträckning av synonymer, dialektala eller syntetiska ord. Språkliga reformer har fortgått sedan dess och nya ord introduceras fortfarande i stor utsträckning. Efter att republiken Turkiet hade bildats och efter alfabetsreformen bildades Turkiska språkkommittén (Türk Dil Kurumu - TDK) under Mustafa Kemal Atatürks beskydd 1932, med syftet att bedriva forskning om turkiskan. En av kommitténs uppgifter var att påbörja en språkreform för att byta ut lånord av arabiskt och persiskt ursprung mot turkiska motsvarigheter. Genom att förbjuda användning av ersatta lånord i pressen lyckades kommittén rensa ut flera hundra ord med icke-turkiskt ursprung ur språket. De flesta ord som infördes i språket av TDK var nya avledningar från turkspråkiga rötter, men TDK valde också att återuppliva fornturkiska ord som inte hade använts på århundraden. På grund av denna plötsliga förändring av språket började äldre och yngre personer i Turkiet skilja sig i fråga om det ordförråd de använde. Medan de generationer som föddes före 1940-talet tenderar att använda äldre termer av arabiskt eller persiskt ursprung, föredrar de yngre generationerna nya uttryck. Exempelvis använde Atatürk själv i sitt stora tal till parlamentet 1927 en osmansk talarstil som idag låter så främmande att man har varit tvungen att "översätta" den två gånger till nutida turkiska: först 1963, sedan 1986. Det finns också en politisk dimension i språkdebatten, då konservativa grupper tenderar att använda mer arkaiska ord i pressen eller i vardagsspråk. Under de senaste årtiondena har TDK fortsatta att mynta nya turkiska ord för att uttrycka nya begrepp och teknologier när de kommer in i språket, mestadels från engelskan. Många av dessa ord, särskilt termer inom informationsteknologi, har blivit allmänt accepterade, men TDK kritiseras emellanåt för att mynta ord som låter påhittade och konstgjorda. Vissa tidigare nyskapade ord, såsom bölem (som skulle ersätta fırka, "politiskt parti"), fick inte allmänhetens gillande. Istället har fırka ersatts av det franska lånordet parti. Vissa ord som återupplivats från fornturkiskan har antagit specialiserade betydelser: till exempel används betik (som ursprungligen betydde "bok") nu med betydelsen skript i datavetenskap. Många av de ord som har myntats av TDK samexisterar med sina äldre motsvarigheter. Detta sker vanligen när ett lånord får en ny betydelse. Exempelvis används ordet dert, som härstammar från det persiska ordet dard (“smärta”), i turkiskan med betydelsen "problem" eller "bekymmer", medan det inhemska turkiska ordet ağrı används för kroppslig smärta. Ibland har lånordet en något annorlunda betydelse än det inhemska turkiska ordet, vilket leder till en situation liknande samexistensen i engelskan mellan germanska och romanska ord. Bland de gamla ord som ersattes finns termer inom geometri, väderstreck, några månadsnamn och många substantiv och adjektiv. Några exempel på nutida turkiska ord och de gamla lånorden är: Turkiskan talas som modersmål av turkar i Turkiet och i den turkiska dias-

poran i runt 30 andra länder. Turkisktalande minoriteter finns i synnerhet i länder som tidigare (helt eller delvis) tillhörde Osmanska riket, såsom Bulgarien, Cypern, Grekland (främst i Västra Thrakien), Makedonien, Rumänien och Serbien. Över två miljoner turkisktalare bor i Tyskland, och det finns stora turkiskspråkiga grupper i Nederländerna, Österrike, Belgien, Schweiz och Storbritannien. Alla etniskt turkiska invandrare talar dock inte turkiska flytande. Antalet modersmålstalare i Turkiet är omkring 60-67 miljoner, vilket motsvarar ungefär 90-93 procent av befolkningen, och 65-73 miljoner modersmålstalare finns över hela världen. Turkiska talas som första eller andra språk av nästan alla invånare i Turkiet, medan kurdiska är första språk för större delen av de övriga (omkring 3,950,000 enligt uppskattningar 1980). De flesta språkliga minoriteter i Turkiet är dock tvåspråkiga och talar turkiska som andra språk upp till flytande nivå. Turkiskans vidsträckta utbredning beror till stor del på Osmanska rikets erövringar i Mellanöstern och på Balkan samt på den senare turkiska arbetskraftsinvandringen till europeiska länder, främst Tyskland. Turkiska är det officiella språket i Turkiet och är ett av de officiella språken på Cypern. Det har också officiell ställning i Prizrendistriktet i Kosovo samt i flera kommuner i Makedonien, beroende på koncentrationen av turkiskspråkiga i lokalbefolkningen. Turkiska språkkommittén (Türk Dil Kurumu eller TDK) är i Turkiet den myndighet som kontrollerar det turkiska språket. Kommittén har stort inflytande och har sedan den bildades 1932 av Kemal Atatürk under namnet Türk Dili Tetkik Cemiyeti ("Sällskapet för forskning om turkiska språket") låtit genomföra åtskilliga inte alltid helt okontroversiella reformer av det turkiska språket. Turkiska språkkommittén var influerad av språklig purism, och en av dess främsta målsättningar var att ersätta lånord och utländska grammatiska konstruktioner med motsvarigheter av turkiskt ursprung. Dessa förändringar, tillsammans med införandet av det nya turkiska alfabetet 1928, formade den nutida turkiska som talas idag. TDK blev ett självständigt organ 1951, då man avskaffade kravet på att utbildningsministern skulle vara ordförande. Denna ställning fortsatte till augusti 1983, då TDK åter blev ett statligt organ i 1982 års turkiska författning efter militärkuppen 1980. Turkiskan är ett språk med ganska stora dialektala skillnader. Framför allt varierar uttalet av konsonanterna c, ç, ğ, h, k, s, ş och z. Istanbulturkiska är etablerat som Turkiets officiella standardspråk. Trots det utjämnande inflytandet från standarden som används i massmedier och utbildning sedan 1930-talet finns den dialektala variationen kvar. Forskare från Turkiet hänvisar ofta till turkiska dialekter som ağız eller şive, vilket medför en tvetydighet med det lingvistiska begreppet accent, vilket också täcks av dessa ord. Projekt som undersöker turkiska dialekter utförs av flera universitet, samt av en särskild arbetsgrupp i Turkiska språkföreningen, och för närvarande genomförs arbete med att samla och publicera forskningen som en uttömmande dialektatlas över turkiskan. Turkiskans standarddialekt är İstanbul. Rumelice (rumeliska) talas i Rumelien på Balkan och av invandrare därifrån, och inbegriper de distinkta dialekterna i Deliorman, Dinler and Adakale som är influerade av Balkanspråkförbundet. Kıbrıs är namnet på cypriotisk turkiska, som talas av turkcyprioter. Edirne är dialekten i Edirne i Trakien. Ege talas i området vid Egeiska havet och används även i Antalya. De nomadiska Yörük-stammarna i Medelhavsområdet och Balkanhalvön har också sin egen turkiska dialekt.[källa behövs] Güneydoğu talas i sydöst, öster om Mersin. Doğu, en dialekt i Östanatolien, bildar ett dialektkontinuum med azeriska, särskilt med Karapapakdialekter i vissa områden. I regionen Centralanatolien talas Orta Anadolu.[källa behövs] Karadeniz, som talas i den östliga Svarta havsregionen och främst företräds av dialekten i Trabzon, uppvisar substratinflytande från grekiska i fonologi och syntax. Kastamonu talas i Kastamonu och dess omgivningar. Dialekten Hemşince talas av den västra gruppen av hamshenier omkring Rize och är influerad av armeniska. Karamanlıca talas i Grekland, där den också kallas Καραμανλήδικα (Karamanlidika).

Den är karamanlidernas skriftliga standard.[källa behövs] Andra turkiska dialekter är danubiska, eskişehir (i provinsen Eskişehir i västra Anatolien), razgrad, karamanska (i provinsen Karaman i centrala Anatolien), gaziantep (i provinsen Gaziantep i södra Turkiet), urfa (i provinsen Şanlıurfa i sydöstra Turkiet), goynuk (runt en by i Bolu).[källa behövs] Flera av de turkiska konsonanterna har främre och bakre allofoner; artikulationsstället varierar något beroende på om den efterföljande vokalen är främre eller bakre (till exempel [l] före främre vokaler men [ɬ] före bakre). Fonemet /ɣ/, vanligtvis kallat "mjukt g" (yumuşak ge), skrivet 'ğ', har ett mycket speciellt värde. Mellan två främre vokaler representerar det egentligen en ganska svag främre velar approximant, [ɰ], men kan även kontraheras till en palatal, [j]. Då den är i slutet av ord eller före en konsonant förlängs den föregående vokalen. I övriga positioner uttalas den inte alls. Dialektalt förekommer många varianter, däribland att den uttalas [ʁ] eller [ɣ] efter bakre vokal och [g] efter främre, samt ibland även [x]. Ljuden [c], [ɟ] och [l] är i ursprungligen turkiska ord allofoner i komplementär distribution med [k], [g] och [ɬ]. De förra förekommer med främre vokaler och de senare med bakre vokaler. I stavningen skrivs båda serierna <k>, <g> och <l>. I vissa lånord kan dock [c] och [ɟ] förekomma med bakre vokaler, till exempel kâr [caɾ] "vinst" gentemot kar [kaɾ] "snö". När en vokal läggs till substantiv som slutar med postvokaliskt <k> blir <k> <ğ> genom konsonantalternation. Turkiskans vokaler är, i alfabetisk ordning, a, e, ı, i, o, ö, u, ü. Det förekommer inga diftonger i turkiskan och när två vokaler möts, vilket sker sällan och endast i lånord, behåller varje vokal sitt individuella ljud. Ett typiskt drag i turkiskan är vokalharmonin, vilket innebär att alla vokaler i ett ord måste överensstämma med varandra i uttalsposition. Om den första vokalen i ett ord är främre blir i regel även övriga vokaler främre. Detta är särskilt påtagligt i ändelser vars vokaler alltid är antingen främre eller bakre beroende på ordstammens vokal. Det turkiska vokalsystemet kan betraktas som tvådimensionellt, där vokaler kännetecknas av två egenskaper: främre/bakre och rundad/orundad. Vokalharmoni är den princip enligt vilken ett inhemskt turkiskt ord innehåller antingen uteslutande bakre vokaler (a, ı, o, u) eller uteslutande främre vokaler (e, i, ö, ü). Vokalmönstret visas i tabellen nedan. Grammatiska affix har "en kameleontliknande beskaffenhet", och följer någon av följande vokalharmonimönster: Följande exempel, som utgår från kopulan -dir4 ("[det] är"), illustrerar vokalharmonins principer i praktiken: Türkiye'dir ("det är Turkiet"), kapıdır ("det är dörren"), men gündür ("det är dagen"), paltodur ("det är kappan"). Det finns några undantag från reglerna om vokalharmoni. I sammansatta ord behöver inte vokalerna harmoniera mellan de ord som utgör delar av sammansättningen (alltså är former som bu|gün ("idag") och baş|kent ("huvudstad") tillåtna). Dessutom tillämpas inte vokalharmoni för lånord och vissa oföränderliga affix, såsom -yor (presens) och -bil- (potentialis). Vissa lånord uppvisar dock partiell eller fullständig vokalharmoni (till exempel mümkün "möjlig" < arabiska mumkin, och dürbün "kikare" < persiska dūrbīn). Det finns också några få inhemska turkiska ord som inte följer regeln, såsom anne ("moder"). I sådana ord - och i lånord - harmonierar suffixen med den sista vokalen: alltså annedir ("hon är mor"). Vägskylten på bilden ovan illustrerar alla dessa egenskaper: Betoningen är vanligtvis på den sista stavelsen, med undantag för vissa ändelsekombinationer och vissa fåstaviga ord, som till exempel masa, "bord", som uttalas ['mɑsɑ]. Undantag utgör dessutom vissa lånord, särskilt från italienska och grekiska, liksom många egennamn. Sådana lånord har oftast betoning på den näst sista stavelsen (/ło'kanta/ lokanta "restaurang" och /is'kele/ iskele "kaj"), men betoningen av egennamn är mindre förutsägbar (/is'tanbuɬ/ İstanbul, /'aŋkaɾa/ Ankara). Turkiskan är ett agglutinerande och vokalharmoniserande språk som ofta använder affix, särskilt suffix (ändelser). Ordföljden är subjekt–objekt–predikat. Suffix används vid ordbildning samt för att indikera den grammatiska funktionen hos ett ord. De kan uttrycka ett ords kasus

och ägarens person, vilket gör att ett ord kan fylla en hel menings funktion (till exempel evinizdeyiz, "vi är vid ert hus"). Agglutinationen innebär att många suffix läggs på ordstammarna vilket innebär att mycket långa ord kan bildas. I stort sett kan hur många suffix som helst fogas på ett ord efter varandra. Suffix kan också användas för att skapa nya ord, såsom att skapa ett verb från ett substantiv, eller ett substantiv från en verbrot (se avsnittet om ordbildning). De flesta affix anger ordets grammatiska funktion. Prefix är dock betydligt ovanligare. De enda inhemska prefixen är allitererande intensifierande stavelser som används med adjektiv eller adverb: till exempel sımsıcak ("kokhet" < sıcak) och masmavi ("klarblå" < mavi). Prefix förekommer även i lånord. Den vidsträckta användningen av affix kan ge upphov till långa ord. Det sägs skämtsamt att det längsta turkiska ordet är Çekoslovakyalılaştıramadıklarımızdanmışsınız, som betyder "Ni sägs vara en av dem som vi inte lyckades omvända till tjeckoslovak". Detta exempel är förstås påhittat, men långa ord förekommer faktiskt ofta i normal turkiska, som i denna rubrik till en dödsannonskolumn i en tidning: Bayramlaşamadıklarımız (Bayram [festival]-recipr-impot-partic-plur-posspl1; "De av oss med vilka vi inte kan utbyta bayramhälsningar"). Turkiskan saknar genus. Detta innebär att det till exempel endast finns ett tredje persons personligt pronomen, o, som är oberoende av den tillsyftades kön (och som alltså kan översättas till svenska som både "han", "hon" och "den"/"det"). Likaså saknas genusvarianter på flertalet substantiv; dessa specificeras vanligtvis genom ett genusbestämmande adjektiv (till exempel erkek çocuk, "pojke", egentligen "manligt barn"). Det finns ingen bestämd artikel i turkiskan, men bestämdhet hos objektet impliceras när ackusativändelsen används (se nedan). Den obestämda artikeln är bir. Pluralis bildas genom tillägg av suffixet -lar respektive -ler (beroende på föregående stavelses vokal). Pluralmärket -ler2 följer omedelbart på substantivet före kasus- eller andra suffix (till exempel köylerin "byarnas"). Turkiska substantiv böjs genom att de tillförs kasusändelser, liksom i exempelvis latin. Det finns sex substantivkasus i turkiskan: nominativ (grundform), genitiv (ägande), ackusativ (direkt objekt), dativ (indirekt objekt, riktning mot), lokativ (läge eller plats) och ablativ (riktning från). Alla dessa ändelser följer vokalharmoni (visas i tabellen med hjälp av den upphöjda notationen). Böjningen av ağaç illustrerar två viktiga drag i turkisk fonologi: att konsonanter assimileras i suffix (ağaçtan, ağaçta) och att slutkonsonanter blir tonande framför vokaler (ağacın, ağaca, ağacı). Kasusmärket för ackusativ används bara för bestämda objekt; jämför ağaç gördük "vi såg ett träd" med ağacı gördük "vi såg trädet". Pluralmärket -ler2 används inte när en klass eller kategori avses: ağaç gördük kan lika gärna betyda "vi såg träd [när vi gick genom skogen]"—i motsats till ağaçları gördük "vi såg träden [i fråga]". Dessutom kan substantiv ta suffix som tilldelar person: exempelvis -imiz4, "vår". Med tillägg av kopulan (till exempel -im4, "jag är") kan fullständiga meningar bildas. Den interrogativa partikeln mi4 följer omedelbart det ord som frågan gäller: köye mi? "[på väg] till byn?", ağaç mı? "[är det ett] träd?". De turkiska personliga pronomenen i nominativ är ben (1s), sen (2s), o (3s), biz (1pl), siz (2pl, eller formellt/artigt 2s) och onlar (3pl). De böjs regelbundet med några undantag: benim (1s gen.); bizim (1pl gen.); bana (1s dat.); sana (2s dat.); och de oblika formerna av o använder roten on. Alla andra pronomen (reflexiv kendi och så vidare) böjs regelbundet. Turkiska adjektiv böjs inte. De flesta adjektiv kan dock även användas som substantiv, i vilket fall de böjs: exempelvis güzel ("vacker") → güzeller ("(de) vackra personer(na)"). Adjektiv som används attributivt föregår de substantiv som de utgör bestämning till. Adjektiven var ("befintlig") och yok ("obefintlig") används i många fall där svenskan skulle använda "det finns" eller "har", till exempel süt yok ("det finns ingen mjölk", bokstavligt "mjölk(en) (är) obefintlig"); konstruktionen "substantiv 1-GEN substantiv 2-POSS var/yok" kan översättas "substantiv 1 har/har inte substantiv 2"; imparatorun elbisesi yok "kejsaren har inga kläder" ("kejsare(n)s

kläder-hans obefintlig(a)"); kedimin ayakkabıları yoktu ("min katt hade inga skor", bokstavligt "katt-min-s sko-plur.-dess obefintlig(a)-dåtid"). Som förstärkning kan den första stavelsen i ett ord dubbleras (till exempel kan beyaz, "vit", bli bembeyaz, "mycket vit", och sıcak, "varm", kan bli sımsıcak, "mycket varm"). Turkiska verb uppvisar en mångfald av tempus, modus och aspekt. Verben anger person. De kan göras negativa, potentiala ("kan") eller impotentiala ("kan inte"). Dessutom anger turkiska verb tempus (presens, preteritum, inferentialis, futurum och aorist), modus (konditionalis, imperativ, necessitativ och optativ) och aspekt. Negation uttrycks med infixet -me2- omedelbart efter verbstammen. Alla turkiska verb konjugeras på samma sätt, förutom det oregelbundna och defekta verbet i-, turkiskans kopula, som kan användas i sammansatta former (den förkortade formen kallas enklitisk): Gelememişti = Gelememiş idi = Gelememiş + i- + -di Turkiska har flera particip, däribland presens (med ändelsen -en2), futurum (-ecek2), preteritum (-miş4) och aorist (-er2 eller -ir4). Dessa former kan fungera antingen som adjektiv eller substantiv: oynamayan çocuklar "barn som inte leker", oynamayanlar "de som inte leker"; okur yazar "läsare-skrivare = läs- och skrivkunnig", okur yazarlar "läs- och skrivkunniga". Participens viktigaste funktion är att bilda modifierande fraser motsvarande de relativsatser som finns i de flesta europeiska språk. De particip som används i dessa konstruktioner är futurum particip (-ecek2) och en äldre form (-dik4), som täcker både presens- och dåtidsbetydelser. Användningen av dessa "personliga" eller "relativa" particip illustreras i följande tabell, där exemplen visas enligt det grammatiska kasus som skulle ses i motsvarande relativsats i svenskan. Ordföljden i enkla turkiska meningar är i allmänhet subjekt objekt predikat, som i japanska och latin, men till skillnad från svenska och engelska. I mer komplexa meningar är grundregeln att bestämningen föregår det bestämda. Den princip inkluderar, som ett viktigt särskilt fall, participbestämningarna som beskrivs ovan. Det bestämda föregår det obestämda: därmed çocuğa hikâyeyi anlattı "hon berättade historien för barnet", men hikâyeyi bir çocuğa anlattı "hon berättade historien för ett barn". Det går att ändra ordföljden för att betona vikten av ett visst ord eller en viss fras. Huvudregeln är att ordet före verbet har betoningen utan undantag. Om man till exempel vill säga "Hakan gick till skolan" och betona ordet "skola" (okul) skulle det bli "Hakan okula gitti". Om betoningen ska läggas på "Hakan" (subjektet), skulle det bli "Okula Hakan gitti", vilket betyder "det var Hakan som gick till skolan". Den osmanska turkiskan hade, genom sina omfattande lån från persiskan och arabiskan, ett mycket stort ordförråd, rikt på synonymer från de tre språken. Dessa lånord utgjorde omkring 20% av den dåvarande turkiskans ordförråd. Under det osmanska rikets sista århundrade lånades allt fler ord från de västerländska språken, såsom franska, engelska och tyska, i och med att den europeiska kulturen hade större genomslagskraft än den traditionella arabiska vid sultanens hov. I och med språkreformen på 1920-talet avskaffades det stora flertalet av de arabiska och persiska lånorden; de ersattes av dialektala, arkaiska och syntetiska ord, men även av synonymer, vilket gjorde turkiskan något ordfattigare. Arabiska och persiska lånord förekommer fortfarande, till exempel cami, "moské", av arabiskans jāmi‘. Efter språkreformen är nylånen företrädesvis tekniska eller kulturellt betingade benämningar, främst från franska (till exempel duş av douche, "dusch") och engelska (till exempel sandviç av sandwich, "smörgås"; futbol av football, "fotboll"). 2005 års upplaga av Güncel Türkçe Sözlük, den officiella turkiska ordboken som ges ut av Turkiska språkkommittén, innehåller 104-481 uppslagsord, av vilka ungefär 14% är av främmande utsprung. Bland de mest betydande bidragsgivarna till turkiskans ordförråd är arabiska, franska, persiska, italienska, engelska och grekiska. Ordbildning sker i turkiskan oftast genom avledning, det vill säga genom att lägga till suffix. I stort sett alla substantiv kan avledas till adjektiv. De flesta turkiska ord har tillkommit genom att avledningsstuffix har förts till ord från

ett relativt litet grundläggande ordförråd. Exempel på en uppsättning ord som avletts från en substantivrot: Ett annat exempel, som utgår från en verbrot: Nya ord bildas också ofta genom sammansättning av två befintliga ord till ett nytt, liksom i svenskan. Några exempel på sammansatta ord ges nedan: Turkiska skrivs med turkiska alfabetet, en modifierad version av latinska alfabetet som infördes 1928 av Atatürk för att ersätta det gamla arabisk-baserade osmanska alfabetet. Det osmanska alfabetet angav endast tre olika vokaler – långa ā, ū och ī – och innehöll flera redundanta konsonanter såsom varianter av z (vilka var åtskilda i arabiskan men inte i turkiskan). Att korta vokaler saknades i det arabiska alfabetet gjorde det särskilt dåligt lämpat för turkiskan, som har åtta vokaler. Skriftreformen var ett viktigt steg i den periodens kulturella reformer. Uppgiften att utarbeta det nya alfabetet och välja de nödvändiga modifieringarna för ljud som är särskilda för turkiskan gavs till en språkkommission som bestod av framträdande lingvister, akademiker och författare. Införandet av det nya turkiska alfabetet hade stöd av utbildningscenter som öppnades runtom i landet, samarbete med förlag och uppmuntran från Atatürk själv, som åkte runt i landet och lärde ut de nya bokstäverna till allmänheten. Läskunnigheten ökade dramatiskt. Turkiskan har nu ett alfabet som är lämpat för dess språkljud. Stavningen är i stort sett fonematisk, med en bokstav för varje fonem. Utöver det latinska standardalfabetet finns bokstäverna ç, ğ, ı (som versalt motsvaras av 'I'), İ (som gement motsvaras av 'i'), ş, ö och ü. Vidare ingår inte q, w och x i alfabetet men används i stavningen av namn på andra språk. [1] Se avsnittet om mjukt g under fonologi ovan.

Abstract

Turkiska (Türkçe) är ett turkspråk som talas som modersmål av ungefär 62 miljoner människor, främst i Turkiet där det är officiellt språk liksom bland turkcyprioterna på Cypern. Turkiska talas av mindre grupper i de delar av Europa som fram till Första världskriget tillhörde Osmanska riket – dvs i Grekland, Bulgarien, och i länder som ingick i det forna Jugoslavien (främst Makedonien och Bosnien). Turkiska talas dessutom av flera miljoner invandrare i Västeuropa, särskilt i Tyskland – så många att det bl.a. ger utslag i Tysklands röster på de turkiska bidragen till Eurovision Song Contest. Det är det mest talade av turkspråken (de turkiska språken). För att inte förväxlas med dessa kallas turkiska ibland Turkiet-turkiska (Türkiye Türkçesi). Turkiskan använder sedan 1920-talet det latinska alfabetet. Detta kan ses som en del i Turkiets allmänna strävan att få betraktas som en del av Europa och av Västvärlden. Språkets rötter kan spåras till Centralasien och de första skriftliga lämningarna är närmare 1200 år gamla. Osmanskan, den omedelbara föregångaren till dagens turkiska, spreds åt väster när det osmanska riket utvidgades. Som en av landsfadern Atatürks reformer när den nya turkiska republiken ersattes 1928 det osmanska alfabetet (en variant av arabiska alfabetet) av en variant av det latinska alfabetet. Samtidigt inledde den Turkiska språkkommittén en arbete för att reformera turkiskan genom att avlägsna alla persiska och arabiska låneord till förmån för turkiska synonymer, och om sådana inte fanns genom nybildningar från turkiska rötter. Typiskt för turkiskan är dess vokalharmoni och utpräglat agglutinerande språkstruktur med många suffix, vilket innebär att mycket långa ord kan bildas. (Andra agglutinerande språk är t.ex. finska, ungerska, swahili och esperanto. Grundordföljden i turkiskan är subjekt objekt verb. Turkiskan har inget grammatiskt genus.

C.7 Wikipedia: f53f45e50ae751e3309b315c5edb8435

Document contents

De stora dragen och alla viktigare händelser i Thomas av Aquinos liv är kända, men biografier skiljer sig i vissa detaljer och datum från varandra. Döden förhindrade Henrik Denifle från att slutföra sitt projekt att skriva en analytisk biografi över Thomas. Denifles vän och elev, Dominic Prümmer, O.P. som var professor i teologi vid universitetet i Fribourg i Schweiz, upptog arbetet och publicerade Fontes Vitae S. Thomae Aquinatis, notis historicis et criticis illustrati. Den första faskikeln, Peter Calos biografi, utkom 1911 i Toulouse. Av den med Thomas samtide Bartolomeus av Lucca får vi veta att tidpunkten för Thomas födelse saknas vilket ledde till en osäkerhet om dennes exakta ålder . Slutet av år 1225 brukar generellt anses vara tiden för hans födelse. Fader Prümmer, som förlitar sig på Calos auktoritet, bedömer 1227 vara en mer sannolik datering. Samtliga är överens om att han avled 1274. Landulph, hans far, var greve av Aquino; Theodora, hans mor, grevinna av Teano. Hans släkt var befryndad med kejsarna Henrik VI och Fredrik II, och med kungarna av Aragonien, Kastilien, och Frankrike. Calo berättar om en legend där en helig eremit förutsade hans karriär, och eremiten skall ha sagt till Theodora före nedkomsten: "Han kommer att inträda i predikareorden, och så omfattande kommer hans lärdom och så stor hans helighet att vara att intill denna dag finns ingen som kan mäta sig med honom". Vid fem års ålder, enligt den traditionella dateringen för hans födelse, sändes han för att få sin första skolning av benediktinerna vid Monte Cassino. Flitig i sina studier blev han också tidigt uppmärksammad för sin meditativa personlighet och sin hängivenhet i sina böner, och hans läromästare överraskades av att återkommande få höra barnet fråga "Vad är Gud?" Omkring 1236 sändes han till universitetet i Neapel. Calo berättar att denna förändring skedde genom att Monte Cassinos abbot i egen hög person skrev till Thomas far, att en sådan anmärkningsvärd begåvning som pojken hade inte skulle gömmas bort i det fördolda. I Neapel var Pietro Martini och Petrus Hibernus hans lärare. Enligt krönikörerna överträffade han snart Martini i grammatik, och därför överlämnades hans undervisning till Peter av Irland, som undervisade honom i logik och naturvetenskap. Tidens sed delade de fria konsterna i två delar: Trivium, innehållande grammatik, logik, och retorik; Quadrivium, inbegripande musik, matematik, geometri, och astronomi. Thomas kunde återge läroövningarna med mera djup och klarhet än hans lärare uppvisade. Någon gång mellan 1240 och augusti 1243 inträdde Thomas av Aquino i dominikanorden, som han drogs till genom Johannes av St. Julian, en framstående predikant vid konventet i Neapel. Staden förundrades över att en ung aristokrat som han skulle ikläda sig en tiggarordens munkkåpa. Med blandade känslor skyndade hans mor till Neapel för att träffa honom. Dominikanerna blev då rädda att hon skulle föra bort honom. De sände honom därför till Rom, i avsikt att därifrån skicka honom vidare till Paris eller Köln. Thomas bröder, som var soldater hos kejsar Fredrik, tillfångatog honom på Theodoras begäran utanför staden Acquapendente, och familjen höll honom fången i San Giovannifästningen i Rocca Secca. Där kvarhölls han i nästan två år under vilket hans föräldrar, bröder och systrar på olika sätt försökte sätta käppar i hjulet för hans kallelse. Bröderna försökte snärja honom genom att överlista honom i hans dygdighet, men novisen fördrev fresterskan från sitt rum med ett brinnande vedträ som han grep från eldstaden. Mot slutet av sitt liv bekände Thomas för sin vän och följeslagare, Reginald av Piperno, hemligheten att han, enligt egen utsago, skulle ha emottagit en stor ynnest vid den tidpunkten. När fresterskan hade motats ut ur rummet knäföll han och bad uppriktigt till Gud om att få behålla sitt sinnes och sin kropps integritet. Han föll då i sömn, och, under det att han sov, såg han två änglar

som försäkrade honom om att han blivit bönhörd. Sedan omgjordade de honom med en vit gördel och sade "Vi förlänar dig gördeln av den eviga oskulden." Och från den dagen skulle han inte ha upplevt den minsta frestelse. Tiden Thomas av Aquino tillbringade i fångenskap var inte förgäves. Hans mor gav efter något efter att den första ilskan och sorgen lagt sig; dominikanerna tilläts undervisa honom ytterligare, och hans systrar försåg honom med några böcker — Bibeln, Aristoteles' Metafysiken, och "Sententia" av Petrus Lombardus. Efter arton månader eller två år i fångenskap, antingen genom hans moders tanke att eremitens profetia skulle gå i uppfyllelse eller att hans bröder fruktade Innocentius IV:s eller Fredrik II:s hot, frisläpptes han till dominikanerna som gladdes åt att han under sin bortavaro "gjort lika mycket framsteg som om han hade gjort i studium generale". Thomas avgav omedelbart sina klosterlöften, och hans överordnade sände honom till Rom. Innocentius IV rannsakade grundligt hans motiv till att inträda i predikareorden och sände iväg honom med sin välsignelse och förbjöd all fortsatt inblandning i hans kallelse. Johannes Teutonikern, ordens fjärde övergeneral (1240-1252), tog den unge studenten till Paris och, enligt de flesta biografier över honom, till Köln, dit han anlände 1244 eller 1245. Där hamnade han under Albertus Magnus, den ryktbaraste professorn i denna orden. I lärosätena misstolkades Thomas ödmjukhet och fåordighet som dumhet, men när Albert hade åhört hans försvar av en svår avhandling förklarade han: "Vi har kallat denne unge man en dum åsna, men hans gnäggande av doktrinerna kommer en dag att ljuda genom världen." År 1245 sändes Albert till Paris, och Thomas var såsom hans student hans följeslagare. Båda återvände till Köln 1248. Albert hade utsetts till ledare av en ny Studium Generale därstädes, utnämnd det året av generalkapitlet, och Thomas skulle föreläsa under honom som magister. Under sin vistelse i Köln, troligen år 1250, prästvigdes han av Konrad av Hochstaden, ärkebiskop i staden. Hur upptagen med den akademiska karriären han än var, predikade han under hela sitt liv Guds ord, i Tyskland, Frankrike, och Italien. Hans predikningar lär ha varit kraftfulla, genomsyrade av fromhet, fulla av handfasta anvisningar, med löpande hänvisningar till de heliga skrifterna. År 1251 eller 1252 sände övergeneralen av orden, efter förslag från Albertus Magnus och Hugo av St. Cher, Thomas till att inta posten som magister (subregent) vid dominikanernas Studium i Paris. Denna utnämning kan betraktas som början på hans offentliga karriär, för hans undervisning tilldrog sig snart uppmärksamheten både av professorerna och av studenterna. Han uppgifter bestod huvudsakligen i att förklara "Sententia" av Petrus Lombardus, och hans kommentarer till den teologiska texten lade grunden till materialet och idén till hans huvudverk, Summa theologica. I vederbörlig tid uppmanades han att förbereda sig på att ta doktorsexamen i teologi vid universitetet i Paris, men universitetets examensrätt drogs tillbaka med anledning av en dispyt mellan universitetet och ordensbröderna. Konflikten, som till en början uppstod mellan universitetet och statliga tjänstemän, blossade upp då en student blev ihjälslagen och tre andra skadades av stadens väktare. Universitetet som slog vakt om sin självständighet, krävde upprättelse vilket avböjdes. Doktorerna stängde stadens skolor, och svor på att inte öppna dem igen förrän deras krav uppnåtts, och bestämde att ingen i framtiden skulle beviljas doktorsgraden som inte svor en ed på att följa deras exempel om de hamnade i liknande omständigheter. Dominikanerna och franciskanerna hade fortfarit att undervisa vid sina skolor och vägrade att avlägga eden, vilket blev upptakten till en bitter strid vars höjdpunkt inträffade när Thomas och Bonaventura skulle doktorera. Vilhelm av St-Amour drev konflikten längre ändå, och attackerade ordensbröderna våldsamt, vilka han uppenbarligen var avundsjuk på, och förnekade dem rätten att inneha lärostolar vid universitetet. I opposition mot dennes bok De periculis novissimorum temporum, skrev Thomas avhandlingen Contra impugnantes religionem, en försvarsskrift för det religiösa ordensväsendet. Vilhelm av St-Amours bok fördömdes av påven Alexander IV i Anagni den 5 oktober 1256, och

påven beordrade att tiggareordens bröder skulle tillåtas att doktorera. Ungefär vid samma tid bekämpade Thomas också en annan bok som Katolska kyrkan bedömde som farlig, "Det eviga evangeliet". Befattningshavarna vid universitetet dröjde innan de tillät Thomas att ta examen; först efter att Ludvig IX av Frankrike använt sitt inflytande och efter elva påvliga uttalanden lade sig konflikten, och Thomas kunde doktorera i teologi. Datumet för hans promovering är enligt många samstämmiga källor den 23 oktober 1257. Ämnet var "Kristi majestät". Verserna "Du vattnar bergen från din sal, jorden mättas av allt du ger" (Psaltaren 104:13), sägs ha varit ett vägledande råd han fick av en himmelsk uppenbarelse. Enligt traditionen skall han och Bonaventura ha doktorerat samma dag, och de båda vännerna skulle ha överträffat varandra i ödmjukhet om vem som skulle promoveras först. Från och med denna tid kan Thomas liv sammanfattas i ett fåtal ord: böner, predikan, undervisning, författande, resande. Människor föredrog snart att lyssna till honom än till Albert. Paris höll fast honom; påvarna önskade honom nära sig; ordens Studia var angelägna om att åtnjuta hans undervisning; följaktligen finner vi honom i Anagni, Rom, Bologna, Orvieto, Viterbo, Perugia, i Paris igen, och slutligen i Neapel, alltjämt undervisande och skrivande, följande sin enda passion i livet att försvara de kristna doktrinerna. Så hängiven var han sin uppgift att han i tårar bönföll att få slippa intaga ärkebiskopsätet i Neapel som han tilldelades av Clemens IV år 1265. Om hans utnämning hade verkställts, hade troligen aldrig Summa theologica blivit skriven. Han gav efter för sina ordensbröders önskan och deltog vid upprepande tillfällen i reformeringen av Kyrkan. Ett av dessa kapitel hölls i London 1263. Under ett annat i Valenciennes 1259 samarbetade han med Albertus Magnus och Peter av Tarentasia (senare påve Innocentius V) i en utformning av ett studiesystem som till sitt innehåll bevarats till våra dagar i dominikanordens Studia Generalia. Det är inte överraskande att kunna läsa i Thomas biografier att han ofta verkade vara frånvarande och i extas. Mot slutet av hans liv uppkom dessa extaser allt oftare. Vid ett tillfälle, i Neapel år 1273, efter att han slutfört sin avhandling om nattvarden, såg tre ordensbröder honom upplyft i en extas och de skall ha hört en röst från altarets krucifix som sade: "Du har skrivit väl om mig, Thomas; vilken belöning önskar du?" Thomas skulle ha svarat: "Ingen annan belöning än dig, Herre". Liknande förklaringar sägs ha gjorts i Orvieto och i Paris. Den 6 december 1273 lade han pennan på bordet och skulle därefter praktiskt taget aldrig mera skriva. Den dagen erfor han en ovanligt lång extas under mässan; vad som uppenbarades honom är ovisst och kan endast utläsas mellan raderna i hans svar till Fader Reginald, när denne hade uppmanat honom att fortsätta skriva: "Jag kan inte göra något mera. Sådana hemligheter har uppenbarats för mig att allt jag hittills har skrivit nu synes mig äga litet värde". Summa theologica hade endast hunnit till den nittonde frågan av den tredje delen (De partibus poenitentiae). Thomas började omedelbart ombesörja förberedelserna för sin död. Gregorius X, som hade sammankallat till Andra Lyonkonciliet 1 maj 1274, bjöd in Thomas och Bonaventura att deltaga i mötet och sade åt den förre att medtaga sin avhandling Contra errores Graecorum till Lyon. Thomas försökte hörsamma befallningen, och började sin färd i januari 1274, men styrkan svek honom; i närheten av Terracina föll han till marken, varpå han fördes till slottet i Maienza, som var hans syskonbarns, grevinnan Francesca Ceccanos, hem. Cisterciensmunkarna i Fossa Nuova övertalade honom att bli deras gäst och han flyttades till deras kloster, dit han anlände viskande till sin följeslagare: "För evigt är detta min viloplats, här vill jag bo, det är min önskan" (Psaltaren 132:14). När fader Reginald uppmanade honom att stanna kvar vid klostret, svarade han: "Om Herren önskar ta mig härifrån är det bättre att han hittar mig i ett religiöst hus än i hägnet av lekmän." Cistercienserna var så vänliga och påpassliga att Thomas besvärades. "Varför föräras mig denna ära", sade han, "då Guds tjänare borde bära ved till mitt bål!" Efter enträgna förfrågningar av

munkarna dikterade han en kort kommentar till Höga visan. Slutet var nära. När sista smörjelsen och sista nattvarden (Viaticum) bars in i rummet uttalade han följande trosbekännelse: Thomas dog den 7 mars 1274. Flera mirakler uppges av Katolska kyrkan ha föregått hans helgonförklaring, och den 18 juli 1323 kanoniserades han av Johannes XXII. Munkarna i Fossa Nuova ville behålla hans kvarlevor men på order av påve Urban V överlämnades hans kropp till det dominikanska brödraskapet, och blev under högtidliga former förd till dominikanernas kyrka i Toulouse den 28 januari 1369. Ett magnifikt relikskrin som uppförts 1628 förstördes under franska revolutionen och kroppen flyttades då till St. Serninkyrkan, där den nu vilar i en sarkofag av guld och silver. Sarkofagen välsignades av kardinal Desprez den 24 juli 1878. Ett ben av hans vänstra arm finns i Neapels katedral. Den högra armen som ursprungligen fanns i dominikanernas Sankt Thomaskapellet, skänktes av universitetet i Paris till dominikanernas kyrka Santa Maria sopra Minerva i Rom, dit den fördes under franska revolutionen. En beskrivning av Thomas såsom han framträdde i livet har Calo givit , som säger att hans utseende överensstämmer med hans stora själ. Han var högväxt och kraftigt byggd, men upprätt och väl proportionerlig. Hans hud hade "samma färg som ungt vete": hans huvud var stort och välformat, och han var något flintskallig. Alla porträtt framställer honom som nobel, meditativ, mjuk, men stark. Pius V utsåg Thomas till kyrkolärare 1567. I encyklikan "Aeterni Patris", från den 4 augusti 1879, vid återställandet av den kristna filosofin, förklarade Leo XIII honom vara "fursten och mästaren av alla skolastiker". Samme pontifikat, genom ett meddelande 4 augusti 1880, utsåg honom till skyddshelgon över alla katolska universitet, akademier, lärosäten och skolor i hela världen. Fastän Thomas bara levde ungefär femtio år, har han skrivit mer än 60 böcker, några kortare och några av avsevärd längd. Detta behöver inte nödvändigtvis innebära att varje ord i de autentiska texterna är skrivna av hans hand; han bistods av sekreterare, och hans biografer uppger att han kunde diktera flera verk samtidigt. Andra verk har felaktigt tillskrivits honom som upphovsman; några av dessa är i stället skrivna av hans lärjungar. I Scriptores Ordinis Praedicatorum (Paris, 1719) tillägnar fader Echard 86 foliosidor åt Thomas arbeten, och åt de olika utgåvorna och översättningarna (I, pp. 282-348). Touron (op. cit., pp. 69 sqq.) säger att avskrifter fanns i nästan alla bibliotek i Europa, och att efter boktryckarkonsten uppfunnits, spreds böckerna snabbt i Tyskland, Italien och Frankrike, och att delar av Summa theologica var ett av de första viktigare verk som trycktes. Peter Schöffer, en boktryckare i Mainz, gav ut Secunda Secundae 1467. Detta är den första kända tryckta utgåvan av Thomas böcker. Den första kompletta utgåvan av Summa trycktes i Basel 1485. Många andra utgåvor av denna och andra verk publicerades på 1500- och 1600-talen, särskilt i Venedig och Lyon. Hans samlade verk (Opera Omnia) utgavs i följande ordning: Rom, 1570; Venedig, 1594, 1612, 1745; Antwerpen, 1612; Paris, 1660, 1871-80 (Vives); Parma, 1852-73; Rom, 1882 (Leonineutgåvan). Den romerska utgåvan från 1570, kallad "Pianan", eftersom den är ett beställningsverk av Pius V, var standardutgåvan i många år. Bortsett från det noggrant redigerade råmaterialet innehåller den kommentarer av kardinal Cajetan och "Tabula Aurea" av Petrus av Bergamo. Venedigutgåvan från 1612 prisades högt eftersom Cajetan-Porrecta-kommentarerna medföljde råmaterialet. Leonineutgåvan tillkom under beskydd av Leo XIII, som gav anvisningar till denna med särskilda Motu Proprio för frågan. Denna innehåller Sylvester Ferrariensis kommentarer till Summa contra gentiles, och Cajetans kommentarer till Summa theologica. De sista banden, IV-XII, av den senare utgåvan utkom 1906. Thomas verk kan klassificeras som filosofi, teologi, bibelvetenskap, apologetik, eller som stridsskrifter (så kallade kontroverser). Klassificeringen upprätthålls dock inte fullt ut i samtliga verk. Summa theologica till exempel, är till sitt innehåll filosofi, medan Summa contra gentiles huvudsakligen, men inte uteslutande filosofi och apologetik. Hans filosofiska verk utgörs i

första hand av kommentarer till Aristoteles, och hans första viktigare teologiska skrift är kommentarer till Petrus Lombardus fyra böcker "Sententia"; men han följer ingen av dessa tänkare slaviskt. Bland de verk där Thomas egenart framträder till tankegods och metod, bör särskilt följande nämnas: Författaren själv beskriver Summa theologica som en lärobok för studenter om de kristna doktrinerna. Det är mera sanningsenligt att beskriva det som en komplett vetenskapligt arrangerad exposition av teologin och på samma gång en sammanfattning (summæ) av den kristna filosofin som världen kände. I det kortfattade förordet påtalar Thomas de svårigheter som hans studenter hade stött på vid studiet av doktrinerna, och han definierar svårigheterna: flertaliga meningslösa anföringar, artiklar och argument; frånvaro av en vetenskaplig ordning; återkommande upprepningar; vilka sammantagna "väcker avsky och förvirring hos de studerande". Sedan tillfogar han: "I önskan att undvika dessa och liknande olägenheter skall vi i förtröstan på att Gud ledsagar oss, bemöda oss om att behandla dessa saker som utreder doktrinerna så kortfattat och så tydligt som ämnet må tillåta." I den inledande frågan, "Om den heliga doktrinen", redogör han för ståndpunkten att vid sidan av den kunskap som kommer av förnuftet är uppenbarelse nödvändig och primär, för det första eftersom utan den skulle inte människan förstå den övernaturliga ände som hon på frivillig väg måste försöka finna; för det andra eftersom de sanningar om Gud som man kan nå genom förnuftet endast kan erhållas av ett fåtal, efter lång tid och genom många felslut och misstag. När den uppenbarade sanningen godtagits fortsätter medvetandet att förklara den och dra slutledningar från den. Därmed är teologin, som enligt honom är en exakt vetenskap, resultatet, eftersom den fortsätter arbeta med de principer som är givna (svar 2). Objektet för vetenskapen är Gud; andra ämnen behandlas endast i så måtto de relaterar till Gud (svar 7). Förnuftet används i teologin inte för att bevisa trons sanning - denna godtas på basis av Guds auktoritet - utan för att försvara, förklara, och utveckla de uppenbarade doktrinerna (svar 8). Den första delen av Summa är indelad i tre traktat: Den andra delen, Om Gud såsom människans mål, kallas ibland Thomas moralteologi, det vill säga hans avhandling om människans mål och om mänskliga handlingar. Den är underindelad i två delar, kända som Första sektionen av den andra (I-II, eller 1a 2ae) och Andra sektionen av den andra (II-II, eller 2a 2ae). Den första sektionen av den andra. De första fem frågorna tillägnas bevisen för att människans ände, hennes salighet, vilar i Guds hand. Människan uppnår det målet eller förlorar den genom sina mänskliga handlingar, det vill säga genom den fria viljans handlingar. Människans handlingar behandlar han först i allmänna ordalag (de första fem frågorna av I-II), och därefter redogör han för specifika fall (II-II). Avhandlingen om mänskliga handlingar i generella ordalag är indelad i två delar: den första redogör för handlingar i sig, och den andra för principerna eller orsakerna, yttre eller inre, till dessa handlingar. I dessa traktat och i Den andra av den andra följer Thomas Aristoteles i sin beskrivning och analys av människans medvetandes och hjärtas rörelse. Den andra av den andra iakttager specifika mänskliga handlingar, det vill säga dygder och synder. I denna behandlar Thomas dels sådant som gäller alla människor oavsett var de befinner sig i livet, dels sådant som endast gäller somliga. Sådant som gäller alla människor är reducerat till sju rubriker: Tro, hopp och kärlek (charitas), vishet, rättvisa, tapperhet och måttfullhet. För att undvika upprepning har Thomas under varje rubrik inte endast behandlat dygden, utan även synden som är dess motsats, påbuden att handla enligt dygden, och den helige andes gåva som svarar emot den. Sådant som gäller endast somliga är reducerat till tre rubriker: den ovillkorliga nåden (gratia gratis datae) som ges till vissa individer till Kyrkans godo, sådant som tungomålstalande, profetians gåva, mirakler, det aktiva och det kontemplativa livet; olika faser i livet, och plikter som tillkommer dem som har olika livsuppgifter, i synnerhet biskopar och religiösa. Den tredje delen behandlar Kristus och vad han har givit människan, och består av

tre traktat: Om inkarnationen och om vad frälsaren utförde och om hans lidande; Om sakramenten, vilka har instiftats av Kristus och vilka verkar genom hans gärningar och lidande; Om det eviga livet, det vill säga världens ände, kropparnas uppståndelse, den yttersta domen, syndarnas straff, och de rättfärdigas salighet vilka genom Kristus får evigt liv i himlen. Åtta år ägnade Thomas åt att sammanställa detta arbete, vilket påbörjades i Rom, där Första delen och Första av den andra skrevs (1265-1269). Den andra av den andra påbörjades i Rom och slutfördes i Paris (1271). 1272 reste Thomas till Neapel där den tredje delen tillkom, till och med den nittonde frågan av tratatet Om botgöring. Detta arbete har slutförts genom en komplettering av ett supplement som utgörs av andra skrifter av Thomas, men vilka somliga tillskriver Petrus av Auvergne, och andra Henrik av Gorkum. Dessa tillskrivanden tillbakavisas av utgivarna av Leonineutgåvan (XI, pp. viii, xiv, xviii). Mandonnet (op. cit., 153) hänger sig åt en vedertagen uppfattning att sammanställningen gjordes av Reginald de Piperno, Thomas trofaste följeslagare och sekreterare. Hela Summa består av 38 avhandlingar, 612 frågor, underindelade i 3120 artiklar, i vilka omkring 10 000 invändningar framförs och besvaras. Summa är den kristna doktrinen i vetenskaplig form; den är det mänskliga förnuftet som tagit tjänst att bevisa sanningen i den kristna religionen. Där är också den vuxne mannen och helgonförklarade kyrkoläraren som ger svar på frågan från barndomen: "Vad är Gud?" Katolska kyrkans vördnad för Thomas och hans Summa är gränslös: "Bland de skolastiska lärarna, tornar Thomas av Aquino som den främste och mästaren av dem alla, eftersom, vilket Cajetan iakttager (i 2am 2ae, Q. 148, a. 4), 'han vördade de forna kyrkolärarna på ett sådant säreget sätt att all deras sammantagna intelligens verkar ha nedärvts i honom'" (encyklikan, "Aeterni Patris", av Leo XIII). Det är omöjligt att beteckna Thomas metod om endast ett ord får användas, så vitt den inte bör kallas eklektisk. Den är aristotelisk, platonsk, sokratisk; den är induktiv och deduktiv, analytisk och syntetisk. Han väljer den metod som för tillfället bäst tjänar hans syften, i en ärlig strävan att erkänna vad som är sant och vederlägga vad som är falskt. Hans styrka därvidlag ligger i att med ett fåtal ord sammanfatta en mängd motstridiga källor och skrifter, och finna gemensamma drag i dessa. För sin tid är Thomas osedvanligt icke-dogmatisk, och kräver inte av läsaren att denne skall förlita sig till hans ord, utan redovisar noggrant för hur slutsatserna dras och hur han kommer fram till sina svar. I filosofi, säger han, är auktoriteters argument av underordnad betydelse; filosofi handlar inte om att veta vad andra har sagt, utan om att känna och förstå sanningen (I lib. de Coelo, lect. xxii; II Sent., D. xiv, a. 2, ad 1um). Filosofin är ett redskap för Thomas, som används för förståelsen av teologin, men han håller fast vid ämnenas givna ramar. Mot traditionalisterna har Heliga stolen förklarat att metoderna som Thomas och Bonaventura använde inte leder till rationalism (Denzinger-Bannwart, n. 1652). Varken hos Albertus Magnus eller Roger Bacon finner man ett motstycke i hans undersökningar av naturen; Thomas var före sin tid vad gäller vetenskap och många av hans åsikter därom har löpande aktualiserats i den akademiska debatten till 2000-talet. Till exempel kan man läsa följande som Thomas skrivit, vilket kan jämföras med sexualsystemet: "I en och samma planta finns en tvåfaldig dygd, aktiv och passiv, fastän det aktiva ibland bara finns i en och det passiva i en annan, så att en planta sägs vara maskulin och den andra feminin" (3 Sent., D. III, Q. ii, a 1). Thomas stil är en blandning av den rakframma framställning som återfinns hos många skolastiker, och den nogräknade stilistiken man kan finna hos Johannes av Salisbury. Påve Innocentius VI (citerad i encyklikan "Aeterni Patris", av Leo XIII) skriver att, med undantag av de kanoniska skrifterna, överträffar Thomas allt skrivet i "uttryckets lämplighet och påståendenas sanning" (habet proprietatem verborum, modum dicendorum, veritatem sententiarum). Många framstående talare har studerat hans retorik. Samma sak gäller teologer. Cajetan var en större kännare av Thomas stil än någon annan. Thomas hade

inte fått denna talets gåva gratis, även om man erkänner att han hade en anmärksningsvärd, medfödd begåvning; han var också strävsam och omarbetade sina skrifter. "Författarens manuskript till Summa Contra Gentiles är till stora delar ofullbordat. Det finns nu i Vatikanbiblioteket. Manuskriptet består av pergamentremsor i skiftande nyanser och färger som förvaras i en gammal pergamenthållare som det förr var fäst vid. Texten är skriven i två kolumner och är svår att tyda, full av förkortningar, och stundom påminner den om stenografi. Genom många stycken har det strukits streck som tyder på raderingar" (Rickaby, Op. cit., förord: se Ucelli ed., "Sum. cont. gent.", Rom, 1878). Hans skolgång var sådan att man kunde förvänta sig att han skulle göra sig ett namn; han erhöll den bästa utbildning som västvärlden kunde erbjuda under 1200-talet, vid Monte Cassino, i Neapel, Paris, och Köln. Han var samtida med flera framstående tänkare: Alexander av Hales, Albertus Magnus, Bonaventura, Raimund av Peñafort, Roger Bacon, Hugo av St Charo, Vincent av Beauvais, och många flera. Framför allt bör Albertus Magnus framhållas, eftersom han var Thomas lärare i Paris och Köln. De böcker som var hans främsta impulsgivare var Bibeln, dekret från koncilier och påvar, de grekiska och latinska kyrkofädernas skrifter, i synnerhet Augustinus, Sententia av Petrus Lombardus, skrifter av de klassiska filosoferna, i synnerhet Aristoteles, Platon och Boethius. Måste man välja några av dessa filosofer som särskilt viktiga för Thomas, vore det Aristoteles, Augustinus och Petrus Lombardus. I ett annat hänseende var Thomas influerad av Averroës, hans huvudmotståndare som han bekämpade för att försvara sin syn på Aristoteles rätta filosofi. Thomas av Aquino har utövat ett ojämförligt inflytande över västvärldens teologi, i synnerhet den romersk-katolska, vilket har spritt sig till den allmänna filosofin där han placerar sig i första ledet av aristotelismen, såsom dess uttolkare, och skolastiken. Ayn Rand (själv ateist) "vidhåller bestämt att Aristoteles var den störste [filosofen] och att Thomas av Aquino var den näst störste." I filosofiskt hänseende är Summa Theologica hans viktigaste och mest livaktiga verk, i vilken han framställer sin systematiska teologi. Thomas ansåg "att vad beträffar kunskap av vad för slag av sanning behöver människan gudomlig hjälp, att intellektet må föras av Gud i denna akt." Han menade dock att människor har en naturlig förmåga att veta många saker utan gudomlig uppenbarelse, även om sådan uppenbarelse sker ibland, "särskilt med hänseende på [sådant som handlar om] tro." Thomas var också en aristoteliker och en empiricist. Dessa två strömningar i västvärldens tänkande gav han ett genomgripande avtryck i. Thomas av Aquino menade att sanning utgörs av förnuftsbaserat vetande (naturlig uppenbarelse) och tro (övernaturlig uppenbarelse). Övernaturlig uppenbarelse uppenbaras genom profeterna, Bibeln, och magisteriet, det som brukar gå under samlingsbeteckningen "traditionen". Naturlig uppenbarelse är den sanning som är tillgänglig för alla människor till följd av deras mänskliga natur; vissa sanningar kan alla människor ernå genom att använda förnuftet på ett korrekt sätt. Exempelvis ansåg han att det mänskliga förnuftet kunde tillgodogöra sig förnuftsbaserade bevis för Guds existens. Fastän Guds väsen och egenskaper (person, enhet, sanning, godhet, makt, kunskap) kan härledas genom förnuftet, kan vissa sanningar endast bli kända genom särskild uppenbarelse (såsom treenigheten). I Thomas teoribildning är den särskilda uppenbarelsen jämförbar med uppenbarelsen av Gud i Jesus Kristus. De överordnade teologiska komponenterna av kristendomen, till exempel treenigheten och inkarnationen, uppenbaras i den Romersk-katolska kyrkans förkunnelse och i Bibeln, och må inte härledas ytterligare. Särskild uppenbarelse (tro) och naturlig uppenbarelse (förnuft) kompletterar snarare än strider emot varandra, för de sammanstrålar i samma punkt: sanningen. En viktig komponent i Thomas filosofi är teorin om analogin. Thomas erkände tre former av beskrivningar: univokation (synonymos), ekvivokation (homonymos) och analogier. Thomas etik bygger på begreppet prima causa. I Summa Theologica, skriver han: Thomas definierade de fyra kardi-

naldygderna som vishet, måttlighet, rättvisa, och mod. Kardinaldygderna är naturliga och uppenbarade i naturen, och de är bindande för samtliga. Det finns dock även tre teologiska dygder: tro hopp och kärlek. Dessa är övernaturliga och åtskilda från de övriga dygderna i deras väsen, vilken är Gud: Thomas urskilde vidare fyra sorters lagar och rätt: evig, naturlig, mänsklig och gudomlig. Den eviga lagen är det Guds ord som styr hans skapelse. Naturrätt och den naturliga lagen är den mänskliga delaktigheten i den eviga lagen vilken människan kommer till insikt om genom förnuftet. Naturrätten följer på prima causa: Viljan att leva och fortplanta sig räknar Thomas till sådana grundläggande (naturliga) mänskliga värden, vilka alla andra mänskliga värden grundas på. Den mänskliga lagen är den positiva rätten: i idealfallet naturrätten tillämpad på samhället av en regering. Gudomlig lag är den lag som i synnerhet tillkännages i de heliga skrifterna. Av den positiva rätten finns två slag: folkrätten och civilrätten. Folkrätten bygger, menar Thomas, på naturrätten och på förnuftiga principer som är förutsättningar för den mänskliga samvaron; dessa är gemensamma för all samhällen. Civilrätten är mer självständig från naturrätten. Thomas har spelat en avgörande roll för den katolska förståelsen av dödssynd och vanesynder. Thomas förnekade att människan har några plikter mot djuren, eftersom de inte är personer. Annars hade det varit i strid emot lagen att äta dem. Detta innebär inte att människan har rätt att behandlade dem med grymhet, för "grymma handlingar kan föras över till hur vi behandlar människor." För Thomas är staten betingad av att människan är en samhällsvarelse. Den är den naturliga ordningen, liksom lagen är naturlig, och så länge som staten inte sätter upp hinder för människornas frälsning, är det följaktligen medborgarnas skyldighet att vara lojal mot staten. Han bygger denna teoribildning på Aristoteles filosofi, och utgör därmed en brytning med Augustinus för vilken staten är ett nödvändigt ont. Ytterst faller statsbegreppet tillbaka på natursynen: naturen och det gudomliga står inte i konflikt för Thomas, varmed heller inte det världsliga gör så av nödvändighet. Till statens naturliga ordning hör också, enligt Thomas, hierarkin, en hierarki som skall bevara den gudomliga ordningen. Men till skillnad från Aristoteles, fördömer Thomas slaveriet. Han företräder en statsteori som bygger på religionen i motsats till den samtide Marsilius av Padua som ville skilja mellan kyrka och stat. Thomas metafysiska lära är aristotelismen på kristen grundval. Från Aristoteles tar han teorin om materia och form, samt om potens och handling (aktualitet och potentialitet). Thomas betraktade teologin, eller "den heliga doktrinen", som en exakt vetenskap, likt naturvetenskapen, där råmaterialet utgörs av de nedtecknade heliga skrifterna och Kyrkans tradition. Dessa källor betraktade han som data, vilka var skapade av Guds självuppenbarelse inför individerna och människorgrupper, löpande genom historien. Tro och förnuft, åtskilda men förbundna till varandra, är två huvudinstrument med vilka teologins data behandlas. Han ansåg att båda var nödvändiga - eller snarare, att sammanstrålningen av båda var nödvändigt - för att erhålla sann kunskap om Gud. Han sammanförde grekisk filosofi med den kristna doktrinen genom att hävda att rationellt tänkande och studiet av naturen, liksom uppenbarelsen, var giltiga sätt att förstå Gud på. Enligt Thomas uppenbarar sig Gud i naturen, varmed studiet av naturen är att studera Gud. Det slutliga målet för teologin, enligt Thomas mening, är att använda förnuftet för att gripa sanningen om Gud och erfara frälsningen genom sanningen. Thomas ansåg att Guds existens varken är självklar (given av sig själv) eller bortom bevisbarhet. I Summa Theologica behandlar han detaljerat fem förnuftsskäl till Guds existens. Dessa kallas quinquae viae, eller "de fem vägarna". Vad beträffar Guds natur ansåg Thomas att det bästa betraktelsesättet, vanligen kallad via negativa, är att utgå från vad Gud inte är. Detta ledde till att han lade fram fem positiva påståenden om de gudomliga egenskaperna: I detta hänseende delade han, bland andra, Maimonides uppfattning om detsamma. Thomas anför att Gud, trots

att han är en fullkomlig helhet, också kan ges en fullständig beskrivning med tre sammanbundna personer: treenigheten. Dessa tre personer (Fadern, Sonen och Den helige ande) konstitueras i och med sina inbördes relationer inom Guds väsen. Fadern skapar Sonen (eller Ordet, Logos) genom självmedvetandet. Denna eviga generation skapar sedan den evige ande "som är bestående av den gudomliga naturen av Guds kärlek, Faderns kärlek till Ordet." Det är inte åtskilt från världen som denna treenighet existerar. Tvärt om tjänar treenigheten till att förmedla Guds själv och Guds godhet till människan. Detta inträffar genom inkarnationen av Ordet (logos) i Jesus Kristus person, och med inblandning av Den helige ande, inom dem som har emottagit Guds frälsning. Thomas börjar sin redogörelse av Jesus Kristus i sin Summa Theologica med att anföra den bibliska berättelsen om Adam och Eva, och beskriva arvssynden. Syftet med Jesu Kristi inkarnation var att återställa den mänskliga naturen genom att avlägsna "den i människan genomsyrade synden," något som människan inte förmår göra på egen hand. "Gudomlig Visdom bedömde det vara passande att Gud skulle bli människa, så att en och samma person både skulle kunna återställa människan och erbjuda tillfredsställelse." Han argumenterar emot flera samtida och historiska teologer som hade en avvikande uppfattning om Kristus. I gensvar till Fotinus säger Thomas att Jesus i sanning var gudomlig och inte enbart en mänsklig varelse. Mot Nestorius, som ansåg att Gud bara bebodde Jesu kropp, anför Thomas att Guds fullkomlighet var en delmängd i Kristi väsen. När Thomas bemöter Apollinaris synsätt, säger han dock att Kristus, också, i sanning hade en mänsklig (förnuftsbaserad) själ. Detta skapar en dualitet i Kristi natur, som är ett synsätt som står i motsättning till Arius läror. Emot Eutyches resonerar Thomas för idén att denna dualitet kvarblev efter inkarnationen. Dessa två naturer existerar simultant fastän oskiljaktiga i den mänskliga kroppen, menar Thomas i strid emot vad Manichaeus och Valentinus hade hävdat. "Kristus hade en riktig kropp av samma natur som våran, en sann, förnuftig själ, och dessutom, fullständig gudomlighet." Därför förefinns både enhet (i en av hans hypostaser) och skiljaktighet (i hans två naturer, den mänskliga och gudomliga) i Kristus. Målet med människans existens är enligt Thomas att förenas med Gud och förevigt leva som hans följeslagare. I synnerhet uppnås detta mål genom den saliga visionen, en händelse genom vilken en person enligt katolicismen erfar fullkomlig, ändlös salighet genom en förståelse av Gud, liksom i Paulus första brev till Korinterna: "Vi ser nu på ett dunkelt sätt, såsom i en spegel, men då skall vi se ansikte mot ansikte. Nu är vår kunskap ett styckverk...". Denna vision, som uppkommer efter döden, är en gåva från Gud som tillkommer dem som har erfarit frälsning och syndernas förlåtelse genom Kristus under deras liv på jorden. Detta ultimata mål ger jordelivets implikationer. Thomas slog fast att en människas fria vilja måste riktas mot det rättrådiga, sådant som välgörenhet, fred och helighet. Han ser detta som en väg till saligheten. Thomas framställning om det moraliska livet baseras på denna idé om salighet. Förhållandet mellan vilja och mål är primära av naturen "eftersom den rättrådiga viljan består av att vederbörligen rätta sig efter änden [det vill säga den saliga visionen]." De som sanningsenligt söker förstå och se Gud kommer av nödvändighet att älska vad Gud älskar. En sådan kärlek kräver moral, och den återgäldas i varje mänskligt val. Att Thomas av Aquino med sina verk utgör medelpunkten i medeltidens tänkande, är allmänt erkänt. För huvudriktningen inom katolicismen råder det numera även konsensus om att det inte sedan Aristoteles funnits någon som utövat större inflytande på tänkandet än Thomas av Aquino; då bör det erinras att stora delar av Europas befolkning är katoliker, liksom folk i andra världsdelar. Hans auktoritet var stor redan under hans samtid. Påvarna, universiteten, studia av hans orden, var samtliga måna om att ta lärdom av hans vetande och vishet. Åskilliga av hans viktigare verk skrevs på beställning, och hans åsikter efterfrågades av flera grupper i samhället. Vid flera tillfällen åberopade doktorerna vid universitetet

i Paris sina disputationer med honom och följde hans anvisningar (Vaughan, op. cit., II, 1 p. 544). Franciskanerna erkände emellertid inte Thomas auktoritet till en början, företrädare för den äldre aristotelismen bekämpade honom med stöd av Augustinus, och en tid stod han för tankar som ansågs som förlegade (via antiqua) i ljuset av bland andra Wilhelm av Ockham (via moderna). Bland Thomas anhängare med störst inflytande finns den så kallade Salamancaskolan, som grundade den moderna folkrätten, folksuveränitetsprincipen, och moderna krigsrätten. Hans principer, som kommer till känna genom hans skrifter, har trots allt fortsatt att influera personer ända till samtiden. Hans betydelse i rent filosofiskt hänseende går under namnet thomism. Av katoliker betraktas han som kristendomens Aristoteles, och det är i det ljuset man skall se hans ställning i påve Leo XIII:s encyklika "Aeterni Patris". Hans betydelse kan sammanfattas i två påståenden: han etablerade katolicismens syn på förhållandet mellan tro och förnuft, och han systematiserade teologin. Thomas principer om förhållandet mellan tro och förnuft fastslogs vid Andra Vatikankonciliet. Det andra, tredje och fjärde kapitlen av den apostoliska konstitutionen "Dei Filius" låter som om sidorna vore tagna direkt från Thomas. Förnuftet ensamt, menar Thomas, är inte tillräckligt för att vägleda människan; människan behöver uppenbarelse; det blir därmed fundamentalt att skilja mellan sanning som erhålls genom förnuftet från sanning som bygger på högre kunskap (mysterier) som kommer till känna genom uppenbarelse. Därtill framhåller han att förnuft och uppenbarelse visserligen är distinkta, men att de inte står i ett motsatsförhållande. Genom tron kan fönuftet räddas från att begå misstag; förnuftet borde tjäna trons syften, menar han. Detta tjänande kan uppträda på tre sätt: Detta är en utveckling av Augustinus tankegångar (De Trin., XIV, c. i). Principerna återfinns på flera ställen i Thomas skrifter, i synnerhet i: "In Boethium, da Trin. Proem.", Q. ii, a. 1; "Sum. cont. gent.", I, cc. iii-ix; Summa I:1:1, I:1:5, I:1:8, I:32:1, I:84:5. Thomas stred inte emot fiktioner, utan gick till angrepp på levande motståndare. Aristoteles verk hade nått Frankrike i form av tvivelaktiga översättningar och kommentarer av judiska och muslimska filosofer, som var oförenliga med den kristna läran. Detta gav upphov till en uppsjö slutledningar bland kristna vilka myndigheterna ansåg vara så alarmerande att Robert de Courçon förbjöd all läsning av Aristoteles Fysiken och Metafysiken år 1210, men dekretet mildrades av Gregorius IX år 1231. Samtidigt uppstod en rationalism vid universitetet i Paris, företrädd av Pierre Abelard och Raymond Lullus, som menade att förnuftet ensamt kunde förklara allt, även trons mysterium, detta i anslutning till Averroës filosofi. Genom att tillämpa Augustinus principer (se I:84:5), och genom att följa Alexander av Hales och Albertus Magnus fotspår, löste Thomas situationen då han lät översätta Aristoteles, bemötte motståndarnas påståenden i stället för att censurera dem, och sökte så att "rena" Aristoteles. Nästa steg var att överföra teologin till en systematisk, vetenskaplig form som tjänade trons syften. Skolastik innebär inte, som somliga påstår, i ändlösa diskussioner och formbundna subtiliteter, utan i att uttrycka doktrinerna i en språkdräkt och form som är klar, korrekt, och koncis. I encyklikan "Aeterni Patris" citerar Leo XIII Sixtus V:s bulla "Triumphantis" (1588), i vilken denne förklarar att för det rätta användandet av filosofin står vi i tacksamhetsskuld till "de nobla begåvningarna som gjorde skolastiken så formidabel mot sanningens fiender", med "den redovisade samstämmigheten av orsak och verkan, ordningen och följdriktigheten liksom arméns disciplin under ett krig, dessa klara definitioner och distinktioner, de kraftfulla argumenten och hängivna diskussionerna genom vilka ljuset skiljs från mörkret, sanningen från det falska, allt blottlägges och barlägges, som om vore falskheten hos heretikerna iklädda moln av undanflykter och felslut". För stora delar av romersk-katolska kyrkan utgör skolastiken den filosofiska guldåldern. Skolastikernas skrifter är de som belyser mörkret och upprättar ordning i kaoset. Där intager Thomas av Aquino ett av de främsta rummen, som efterföl-

jare till bland andra Anselm av Canterbury och Petrus Lombardus. En nytolkning av Thomas av Aquino under 1900-talet, påbjuden av Vatikanen, har kommit att kallas nythomism. Den kännetecknas framför allt av en opposition mot modernismen, men den utgör också grunden för den politiska katolicismens socialpolitik. I så måtto ligger den bland annat till grund för kristdemokratin, men nythomismen placerar sig inte gärna på höger-vänsterskalan. I en syllabus från 1864 av påve Pius IX, fördöms ståndpunkten att skolastikernas metoder och principer inte är tillämpliga i modern tid eller för vetenskapens fortskridande (Denzinger-Bannwart, n. 1713). Encyklikan "Aeterni Patris" av påve Leo XIII framför skäl till "en reform av praktikerna i filosofi genom att återinföra de välkända lärorna av Sankt Thomas av Aquino". Han uppmanar där biskoparna att "återinföra Thomas gyllene visom och sprida den i när och fjärran till den katolska trons försvar och skönhet, till samhällets godo, och för utvecklingen av alla vetenskaperna". På de sidor i encyklikan som direkt föregår detta citat, förklarar påven varför Thomas lära skulle uppnå sådana eftersträvansvärda resultat: Thomas är, säger påven, den store mästaren i att förklara och försvara tron, för hans är "kyrkofädernas och skolastikernas fasta doktrin, som med en sådan klarhet och kraft redogör för trons grundvalar, dess gudomliga ursprung, dess egenartade sanning, för argument som den står emot, fördelar som den skänker mänskosläktet, och dess fullkomliga harmoni med förnuftet, på ett sätt som fullständigt tillfredsställer sinnen som är öppna för övertalning fastän ovilligt och motstridigt". I sig själv betraktas Thomas karriär som ett bevis för att Katolska kyrkan inte motsätter sig förnuftet, om det används på rätt sätt. De sociologiska aspekterna av Thomas framhålls också i encyklikan: "Thomas lära redovisar den sanna innebörden av frihet /—/, den gudomliga makt som all auktoritet kommer av, lagarna och deras kraft, furstarnas faderliga och rättvisa styrelse, åtlydnanden av de största makterna, ömsesidig välgörenhet - om alla dessa och liknande saker; han har en kraft att stå emot principerna hos den nya ordningen vars fara för freden och den allmänna säkerheten är känd" (ibid.). Det onda som drabbat det moderna samhället har tagits upp av densamme påven i brevet "Inscrutabili" från 21 april 1878, och i ett annat om socialism, kommunism, och nihilism ("The Great Encyclicals of Leo XIII", pp. 9 sqq.; 22 sqq.). Påven anför alltid Thomas filosofi som bot mot detta, och som svar på de sociala och politiska problemen i samtiden, för kristna i olika stater och för arbetarklassen (ibid., pp. 107, 135, 180, 208). Att Thomas och de övriga skolastikernas teorier är oförenliga med det moderna samhällets vetenskaper, tillbakavisades av Leo XIII som anförde följande: (a) Skolastikerna motsatte sig inte vetenskapliga undersökningar (b) Undersökning enbart är inte tillräckligt för sann vetenskap. Detta var en vederläggning av modernisternas försök till nyorientering inom teologin. Rationalismen som Thomas mötte i sin samtid, har många gemensamma nämnare med modernitetens rationalism. Även encyklikan "Providentissimus Deus" (18 november 1893) tillägnar Leo XIII Thomas filosofi. Bland de tänkare, utanför kurian, som förde fram nythomismen finns i främsta rummet Jacques Maritain, som medverkade till utformningen av FN:s allmänna deklaration om de mänskliga rättigheterna. I förordet till sin bok om Thomas tillämpbarhet i det moderna samhället (St. Thomas Acquinas, 1930), definierar Maritain nythomismen och redogör för dess politiska, filosofiska och religiösa implikationer. Étienne Gilson, Martin Grabmann, Antonin-Dalmace Sertillanges och kardinal Mercier var ytterligare några framträdande personer under nythomismens tidiga år. Till den nythomistiska rörelsen hör att flera lärosäten och tidskrifter grundades för studiet av Thomas och andra medeltida skolastiker, till exempel lärosätena Institut supérieur de philosophie, Angelicum (Rom), Institut Catholique (Paris), samt underversitetet i Fribourg. Dessutom har nythomismen spelat en framträdande roll vid kontinentala och anglosaxiska icke-konfessionella universitet. Bland övriga tänkare som företrätt nythomistiska tankegånger, och analytisk thomism, finns bland andra

Elizabeth Anscombe, Alasdair MacIntyre och Philippa Foot. Huvudparten av artikeln bygger på översatta bearbetningar och referat av stycken ur flera verk:

Abstract

Thomas av Aquino, latin Thomas Aquinas, född omkring 1225 i Roccasecca i närheten av Neapel, död 7 mars 1274 i Fossanova, var en italiensk teolog och filosof. Thomas av Aquino kanoniserades 1323 och vördas som helgon inom Romersk-katolska kyrkan. Hans minnesdag firas den 28 januari. Thomas benämns Doctor Communis Ecclesiae, kyrkolärare, och betraktas som den främste av de katolska teologerna, vilket bland annat har gett honom det mer specifika tillnamnet Doctor Angelicus. Den teologiska och filosofiska lära han kom att bilda kallas thomism. Han räknas som en av de främsta företrädarna för skolastiken. Thomas av Aquino anpassade Aristoteles filosofiska system till den kristna tron. Aristoteles hade menat att universum var evigt, men att detta hade försatts i rörelse av den första röraren, något som inte samstämde med kristendomens linjära tidssyn, vilken kräver skapelse (därmed även en skapare) och domedag.