Automatic Text Summarization

Master's Thesis
University of Turku
Department of Future Technologies
Computer Science
2018
Pauliina Anttila

TURUN YLIOPISTO
Tulevaisuuden teknologioiden laitos
PAULIINA ANTTILA: Automaattinen tekstin tiivistys
Pro Gradu -tutkielma, 51 s.
Tietojenkäsittelytieteet
Toukokuu 2018

Automaattinen tekstin tiivistäminen on ollut nopeasti kehittyvä tutkimusala viimeisten 70 vuoden ajan. Menetelmät ovat kehittyneet yksinkertaisista heuristiikoista neuroverkkoihin ja syväoppimiseen. Sekä avainlausekkeet tekstistä poimivat että abstraktoivat menetelmät ovat säilyttäneet kiinnostuksensa tähän päivään asti. Tässä tutkielmassa tutkimme automaattisen tekstin tiivistämisen eri menetelmiä ja arvioimme niiden kyvyn tiivistää suomenkielistä tekstiä. Teemme lauseita poimivan tekstin tiivistäjän ja arvioimme, miten se toimii suomenkielisellä uutisdatalla. Arvioimme myös uutisdatan sopivuutta syväoppivan tiivistäjän kouluttamiseen tulevaisuudessa. Saadut ROUGE-metriikat kertovat, etteivät tulokset ole sillä tasolla, mitä tänä päivänä on odotettavissa. Toisaalta laadullinen arviointi paljastaa, että luodut tiivistelmät ovat usein asiapitoisempia kuin datan esimerkkitiivistelmät.

Asiasanat: tekstin tiivistys, uutisdata, avainsanojen louhinta, tiedonlouhinta, luonnollisen kielen käsittely

UNIVERSITY OF TURKU
Department of Future Technologies
PAULIINA ANTTILA: Automatic Text Summarization
Master's Thesis, 51 p.
Computer Science
May 2018

Automatic text summarization has been a rapidly developing research area in natural language processing for the last 70 years. The development has progressed from simple heuristics to neural networks and deep learning. Both extractive and abstractive methods have maintained their interest to this day. In this thesis we will research different methods on automatic text summarization and evaluate their capability to summarize text written in Finnish. We will build an extractive summarizer and evaluate how well it performs on Finnish news data. We also evaluate the goodness of the news data to see can it be used in the future to develop a deep learning based summarizer. The obtained ROUGE scores tell that the performance is not what is expected today from a generic summarizer. On the other hand, the qualitative evaluation reveals that the generated summaries often are more factual than the gold standard summaries in the data set.

Keywords: summarization, news data, keyphrase extraction, extractive, natural language processing, data mining

Contents

List of Abbreviations

AKE	Automatic keyword extraction
ANN	Artificial neural network
CNN	Convolutional neural network
CRISP-DM	Cross-Industry Standard Process for Data Mining
LSA	Latent semantic analysis
NLG	Natural language generation
NLP	Natural language processing
NLU	Natural language understanding
PoS	Part of speech
RNN	Recurrent neural network
RST	Rhetorical structure theory
SVD	Singular value decomposition
tf–idf	Term frequency – inverse document frequency

Chapter 1

Introduction

Automatic text summarization is part of machine learning, natural language processing (NLP) and data mining. It is becoming a popular research area while data grow and there is a demand to process it more efficiently. The aim is to find the core of the given text set and reduce the size while covering the key concepts and overall meaning and avoiding repetition.

Although the interest in summarization has been growing and new approaches are being developed all the time, the task of automatic summarization has still unanswered questions, e.g. how to achieve a deeper understanding of the document topic (natural language understanding, NLU), how to handle long documents, and how to improve the evaluation methods. Getting answers to these questions will take the research in this area further.

In this thesis, we first get to know the field of text summarization by introducing us to the history, applications, and evaluation metrics of summarization systems. Then we explore and compare different methods in earlier automatic text summarization research. We will start from the simplest and oldest that follow simple heuristics, review some graph-based approaches, and move through latent semantic analysis and rhetorical structure

theory towards neural network based approaches that follow the current trend. Methods are screened against their language independence as we are interested their suitability to summarize Finnish texts. Finally, we develop an unsupervised extractive text summarizer to summarize Finnish news articles and evaluate how good it is. We pay special attention to the quality of the data to assess its suitability in future research.

Chapter 2

Background

There are two different types of summarization: the first is to generate a generic summary of a collection of documents and the other is to generate a summary based on a specific query. In this thesis we focus on the former and specify the collection to consist of text documents rather than videos or images. In automatic text summarization, the source text can consist of multiple text documents or only one document. Here we focus on summarizing one article at a time as each article in the used data set is an individual unit and not related the other articles.

2.1 History

Text mining or text analytics has its roots in data mining. Its popularity grew with the need to be able to process unstructured data, as 80% of the data available today is in unstructured form and thus harder to understand and utilize than structured data. The other 20% is summarized in structured form. Also it has been estimated that 80% of the data is in text format which makes automatic processing a crucial task. [1]

To access information from a document, two similar processes were used. Information

retrieval has a logical query to which it aims to find the answer. Information extraction, on the other hand, tries to extract specific information and analyze it. To make those work, summarization process was needed to reduce the size of the text. [1]

In the history of text summarization, the first applications were library catalogs in 1674 and later generating abstracts for research articles in 1898 [1]. At first, the emphasis was on generating summaries that would help to choose the best articles for deeper reading rather than trying to generate summaries that would replace the original text.

The first summarization system was built on the first commercial computer, IBM 701, by Luhn in 1950s and it was based on bag of words technique and counting word frequencies. He extracted frequently occurring words and then gave each sentence a number based on how many frequent words the sentence has. The number presented the significance of the sentence. Then the abstract was formed of the most significant sentences. [2]

A decade later Edmundson [3] introduced new statistical methods on automatic extraction: Cue, Key, Title, and Location methods. The aim with the Cue method is to have a corpus of words whose appearance in a sentence would make the sentence either important, unimportant or irrelevant. The Key method selects the words that appear in the original text more frequently than in the whole corpus being the start for the tf–idf (term frequency – inverse document frequency) method, the Title method takes into account the title and the headings, and Location method the position of the sentences: sentences under headings and first and last sentences of paragraphs and the document are usually more relevant than other sentences. He also emphasized that semantic and syntactic features of the text should be taken into account in the future development of summarizers, e.g. the length of the summary could be determined automatically, Edmundson set it to 25% of the sentences in the original.

Little by little linguistics was taken into account and systems started to handle different word forms with the techniques of NLP. The focus was on extracting, categorizing, and

classifying text. Between 1990 and 2000 machine learning was introduced in NLP to parse sentences into tokens and stemming words into their base forms [1].

So far the research focused solely on words, and computers were not able to understand the semantics of a text document. Text analytics was anyway evolving rapidly and in the next phase the aim moved to understanding the meaning of the text [1]. Currently researchers are still trying to build systems that are able to understand the semantics and pass reading comprehension tests.

2.2 Applications

The means of communication have changed rapidly in the last two decades and many of the things we write or say increase the amount of data in the world. With speech recognition the data that initially was in spoken format ends up in written format and paper documents are converted into electoring form, and that huge amount of data often needs to be pre-processed or otherwise summarized to decrease its size. [1]

The aim in many of the applications of automatic text summarization is to shorten the given text for a human to read. One real life example is summarization of news articles, which frees up time from humans as they don't have to read so much text to grasp the idea of an event. In multi-document summarization, multiple articles on the same topic can be summarized to generate one human-readable summary article without duplicate content. News from multiple sources can that way be combined together to form one summary that has all the relevant information in concise format. Summarizing can also make the text easier to read for children, non-native speakers or dyslexic people.

In addition to news articles, any other text can also be summarized. This is used for example in internet forums to automatically provide the core content of messages written by users. This way long messages can be shrinked to a format that is faster to read.

One example of a real life application is social media site Reddit's[1] TLDR bot (too long; didn't read) which summarizes news posted by users [4]. Blog summarization is another example, although it performs worse than news summarization [5]. The research area of opinion mining has mostly revolved around sentiment analysis but that has also been combined with summarizing customer reviews [6].

Text summarization could also be utilized along search engines: to retrieve relevant documents based on a given query and to produce reports that cover the main content of all of them. This belongs to the practice area of web mining and is very interesting topic due to the unique structure of web.

It can be advantage in information retrieval if keywords can automatically be extracted from source documents. In scientific research it's a good practice to provide keywords that represent the topic of the research article. Automatic summarization techniques can here help researchers to pick the keywords. We talk more about keyphrase extraction in chapter 2.3.

Summarization systems also help humans to write better summaries. System can for example highlight parts in the content that seem important so that the humans can use them to write the end-summary themselves. This is a good way to fasten the summarization process but still generate grammatically correct and well organized coherent text until the natural language generation (NLG) algorithms are as good as humans.

In the standard way the summary is written in the same language as the original text. That is called mono-lingual summarization. In cross-lingual summarization the summary is in other language, and in multilingual summarization the source text consists of several languages which are also used in the summary. [7] This means that the task of automatic summarization can be combined with machine translation or other related tasks as well and it will bring even more relevant applications for summarization.

[1] https://www.reddit.com

2.3 Evaluating summarization systems

The aim in evaluating a NLP system is to see how well the system reaches the goals and fills the requirements. Based on evaluations different systems can also be compared. Usually, evaluation is done by comparing a gold standard (which often is a human made result) to the results of different tasks given to system. As it is practically not possible to manually evaluate many summaries quickly and consistently without bias, automatic evaluation metrics are needed. In this chapter we will compare different aspects of evaluations and then review some most common automatic evaluation methods for automatic summarization systems.

2.3.1 Automatic vs. manual evaluation

Automatic evaluation is an objective evaluation. It uses a pre-built evaluation system that consist of a battery of tests. Those tests can be run on multiple systems and thus compared objectively. Manual evaluation, on the other hand, is more subjective evaluation and needs human input to test the system. Manual evaluation does not need a reference summary like automatic evaluation does, but the drawback is that it is not scalable: humans needs to read both source documents and summaries which is very time-consuming. It is often also challenging for a human to know what information the summary should contain.

Automatic evaluation can also be thought as a quantitative evaluation and manual as a qualitative. If the goal of the summarizer is to generate a coherent and grammatically correct summary which conveys the key points of the text the same way as humans would do, qualitative evaluation is a good addition to the evaluation process.

2.3.2 Intrinsic vs. extrinsic evaluation

Intrinsic evaluation measures the quality of a method without any application. It is usually faster way to see are the improvements in a system potentially good. In practice intrinsic evaluation is done by comparing automatically generated and gold-standard summaries, but it can also be done by testing the fidelity between the two summaries, i.e. is the content similar. ROUGE method explained later in this chapter is an example of an intrinsic evaluation method.

In extrinsic evaluation a system is tested and evaluated in a real-world situation and is commented on performance, utility and usefulness. Only it can tell whether an improvement really helps in a specific task.

2.3.3 Inter-textual vs. intra-textual

Intra-textual evaluation focuses on one summary whilst inter-textual considers the outputs of several systems. Widely used ROUGE metric is inter-textual evaluation method. It is used in NIST's annual Document Understanding Conferences (DUC), where researchers submit their summarization systems [8] and is explained next in this chapter.

2.3.4 ROUGE evaluation method

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) compares summaries by counting the number of n-gram overlaps for automatic generated summaries and human written summaries. It is based on recall and measures how well automatically generated summary covers the content in human-generated summary. ROUGE includes several evaluation methods: ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-S. ROUGE-N is an n-gram recall between an automatic summary and gold standard summary. Rouge-N is computed as follows:

$$ROUGE - N = \frac{\sum_{S\epsilon\{ReferenceSummaries\}} \sum_{gram_n \epsilon S} Count_{match}(gram_n)}{\sum_{S\epsilon\{ReferenceSummaries\}} \sum_{gram_n \epsilon S} Count(gram_n)} \quad (2.1)$$

where n is the length of the n-gram (gramn), and Countmatch(gramn)is the maximum number of n-grams that occur in both automatic summary and gold-standard summary. [9]

ROUGE-L calculates the longest common subsequence (LCS). The point is, that the longer the LCS, the more similar the summaries are. ROUGE-L also works well at sentence level allowing content between matches. That is a disadvantage sometimes, and to improve the ROUGE-L method, ROUGE-W was invented. It calculates a weighted longest common subsequence - thus takes into account the length of consecutive matches. [9]

ROUGE-S (Skip-Bigram Co-Occurrence Statistics) calculates the overlap of skip-bigrams between two summaries. Skip-bigram is any pair of words of a sentence in the order they appear. They can have arbitrary gaps, thus first and second word of a sentence are a skip-bigram as well as any word and last word (although a maximum skip distance can be set). Thus e.g. a 4-word sentence has always 6 bi-grams. ROUGE-S does not require consecutive matches but values same word order. That is, sentences with exactly same words but opposite word order have ROUGE-S score 0. [9]

The studies show that n-gram co-occurrence statistics is a good automatic scoring metric in single-document summarization task [8]. Lin compared the different ROUGE methods with single document DUC data and found out that all of the methods correlated well with human evaluations (ROUGE-1 or ROUGE-2 often performing better than other ROUGE-N variants). The length of the summaries, stemming and stopword removal seemed to affect the correlation. In addition to good correlation with human evaluation the metric has both high recall and precision. [9] Moen et al. [10] found similar results when they

evaluated the evaluation methods on clinical text: human evaluation agreed on the top summarization methods with ROUGE, but considered them a lot better.

What ROUGE methods do not evaluate is grammatical correctness, readability or whether the sentences flow together well. ROUGE is similar to BLEU measure which is used to evaluate automatic translation tools, but BLEU is based on precision which prefers accuracy [8].

Recall is the ratio of overlapping units to the total number of units in the gold standard. It measures how well the automatically generated summary covers the content in the gold standard. Precision is the ratio of overlapping units to the total number of units in the automatic summary. It measures the quality of the automatic summary as it decreases as the number of false positives grows.

$$recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{2.2}$$

$$precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{2.3}$$

F-Score can then be calculated as

$$F - score = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \tag{2.4}$$

F-Score combines recall and precision providing thus a comprehensive evaluation metric.

Chapter 3

Approaches

In general there are two approaches in summarizing text: extraction and abstraction. Extractive approach tries to find the units that explain the meaning of the original text. An unit can be for example a word, a phrase or a whole sentence. The summary is then formed of these units. The length of the summary depends on the compression rate. Abstractive approach uses NLG to write the summary of semantic representation of the original text. It does not repeat the content like the extractive method does, but paraphrases it in shorter size and thus needs comprehensive NLP. The former has been the focus in research and it suits better also for summarizing other types of content than text. The latter generates an output that a human might produce when summarizing text.

In this chapter we introduce six extractive approaches out of which one utilizes neural networks, and one abstractive neural network approach to automatic text summarization in roughly chronological order. Most of the methods are unsupervised and thus can well be utilised with different languages with only some changes in the preprocessing step. Neural network approach is supervised and would need training data in each language it was used in.

3.1 Naive text summarization

Simple text summarization system can be built based on Luhn's work [2]. As a pre-processing step, all the words in the input text document are tokenized and stopwords are removed. Then the word frequencies are counted for remaining words. Stopword removal ensures that uninformative words like *the* and *is* will not affect the frequency distribution.

Once the frequencies are calculated, each sentence in the document gets a score based on how the frequently used words exist in it. The intuition is, that the frequent words are important and important sentences contain important words. Each frequent word can increase the score for a sentence only once, thus the system prefers sentences with a variety of important words, those are more likely to be topic sentences, i.e. the most important sentences that summarize the main ideas.

Frequency calculations can be advanced with tf–idf (term frequency – inverse document frequency) metric if the text to be summarized belongs to a larger corpus (document). The tf–idf value is proportional to the number of word occurrences in the text. But if the occurrences in the corpus increase the tf–idf value decreases as the word is not unique and hereby significant anymore. If this metric is used, it is not necessary to remove stopwords anymore as inverse document frequency handles them.

Equations 3.1, 3.2 and 3.3 define the term frequency (tf) – inverse document frequency (idf).

$$tf(t,d) = \sum_{i \epsilon d}^{|d|} 1\{d_i = t\} \tag{3.1}$$

$$idf(t) = log(\frac{|D|}{\sum_{d \epsilon D}^{|D|} t \epsilon d}) \tag{3.2}$$

$$tf \cdot idf(t,d) = tf(t,d) \cdot idf(t) \tag{3.3}$$

When all the sentences have a score, arbitrary number of most important sentences can be chosen to the summary.

This approach is very simple. By treating the text as disjointed individual sentences, the resulting summary will lack cohesion from sentence to another. A sentence can also be anaphoric, i.e. take its reference and meaning from another sentence. Thus both sentences should be selected even though other one does not qualify. This method also prefers longer sentences over short sentences as they naturally gain bigger scores. The bag-of-words technique, i.e. considering words without their context of preceding and following words is also not an ideal solution.

Topic sentences in writing is an old research topic. Braddock's research [11] published in 1974 concluded that the topic sentences do not appear in a special place in a paragraph. The research has been revisited by Smith, who, on the contrary, noted [12] that the topic sentences most often (in 62% of the cases where topic sentence was explicit) appear at the beginning of a paragraph. Thus the placement of a sentence may also indicate whether it is important or not. The naive approach could thus be improved by adjusting the scoring system based on the placement of the sentences.

In the naive approach the language of the text does not matter as long as the text can be divided into sentences. Languages with many homonyms[1] or polysemes[2] may have worse results as the semantics are not considered with tokens that look the same. In Finnish language, 15% of words are homonymic [13] , which is quite much, but in database text search research the problem has proved to be relatively small [14]. Part of speech (PoS)

[1] two or more words having the same spelling or pronunciation but different meanings and origins

[2] a word having more than one meaning

tagging would make the problem even more smaller.

3.2 Automatic keyphrase extraction

Keyphrase extraction tries to find a selection of words from the free-text document that best represent the document. In addition to text summarization, it is a useful tool in many other NLP tasks, such as text categorization, opinion mining and document indexing [15].

Commonly automatic keyphrase extraction (AKE) system first extracts the candidate keyphrases from a document and then selects the correct keyphrases using either supervised or unsupervised approach.

In a supervised approach a classifier is trained to predict whether a phrase or a word in a document is a keyword. The approach requires good datasets where the keywords are annotated. Thus it is not the easiest and fastest way to get good results. If the dataset has inconsistency, the results may vary. [16] Recently supervised methods have utilised linguistic knowledge, such as PoS information, in the learning process, and the results have become better.

In a unsupervised approach the text is modelled as a graph where the nodes are the terms and edges between them represent relations: how similar the terms are to each other. [16] The aim is to find the underlying structure without any human effort. The edges are then weighted based on their importance, and each node gets a rank based on the weights. Top ranked notes are selected as the keywords. This method is similar to TextRank, which is presented in the next subchapter.

This method has still problems, which Hasan and Ng point out [15]: statistical importance doesn't guarantee that the term represents the document theme, and a term representing the document theme does not necessarily occur frequently in the document. Alheramy

and Walker introduce SemCluster, an unsupervised clustering-based method for AKE, that takes into account the described problems [16].

After preprocessing the data (tokenization, sentence boundary detection, POS tagging, chunking), SemCluster groups similar terms in the text into a cluster based on their meaning. Each cluster represents a topic in the text. This approach uses terms near the cluster centroids as seeds to find phrases that cover the theme of the text. Finally, phrases that contain one or more centroids, are chosen as keyphrases. As the terms that often are manually annotated as keywords are nouns, SemCluster takes only noun phrases into consideration when choosing keywords. [16]

The lengthier the document, the harder it is to find the keyphrases. The structure of the text can help with finding the keyphrases: they are most likely found in the abstract and introduction. [15]

3.3 Sentence extraction

The aim of sentence extraction is to find the most important sentences from the text document. Its disadvantage is the loss of coherence in the generated summary as sentences are picked as they are and relations to other sentences are not considered.

TextRank is a graph-based unsupervised ranking model used in keyword and sentence extraction. [17] The document text is modeled as a graph where the nodes are sentences, the edges represent relations (in this case they can also be called votes). The amount of votes defines the importance of the sentence. Furthermore, the importance of a vote is defined by the importance of the sentence that is voting. The intuition behind it is that an important sentence is highly related to other important candidate sentences, similarly as PageRank algorithm evaluates the importance of web pages using links between the pages; each link is a vote for the linked page.

Let directed graph with set of nodes V and set of edges E be $G = (V, E)$. In PageRank Brin and Page defined the score for a node V_i be:

$$S(V_i) = (1 - d) + d * \sum_{j \epsilon In(V_i)} \frac{1}{|Out(V_j)|} S(V_j) \tag{3.4}$$

where $In(V_i)$ is the set of nodes that point to V_i and $Out(V_i)$ is the set of nodes that the V_i points to. d is a damping factor (value [0, 1], usually set to 0.85) that represents the probability of a random jump to another node. [18]

The original PageRank algorithm assumes that the graph is unweighted, as it is unusual that a web page links to another page more than once. However, in the case of natural language text, text entities may have partial or multiple relations between them, thus it is useful to use a weighted graph. With weighted graph the score is calculated as follows [17]:

$$WS(V_i) = (1 - d) + d * \sum_{V_j \epsilon In(V_i)} \frac{w_{ji}}{\sum_{V_k \epsilon Out(V_j)} w_{jk}} WS(V_j). \tag{3.5}$$

TextRank ranking algorithm starts with assigning arbitrary values to the nodes in the graph. The the score calculation is iterated until a desired error rate, convergence, for any node is achieved. The error rate is the difference between the $S(V_i)$ and the score computed at iteration k, $S^k(V_i)$. Because the real score is not known before the algorithm is run, the error is calculated as the difference between the scores in two consecutive iterations. [17]

The type of relation between nodes is determined by the type of the application where the TextRank algorithm is used. Relation can be e.g. a measure of lexical similarity (how many tokens are shared by the text units), semantic similarity or contextual overlap. In their research, Mihalcea and Tarau [17] used similarity measure as the relation and

counted a content overlap score for each sentence pair.

The content overlap means the number of common tokens in two sentences. Tokens can be restricted to cover only specific types of words, e.g. nouns and adjectives. To avoid preferring long sentences, score can be normalized by dividing it with the sentence lengths. Thus the similarity between sentences S_i and S_j defined as [17]:

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \epsilon S_i \& w_k \epsilon S_j|}{log(|S_i|) + log(|S_j|)} \tag{3.6}$$

Finally the sentences are sorted based on their scores and arbitrary number of top sentences are chosen.

The advantages of TextRank are that is fully unsupervised and extractive method so it does not need a training and testing datasets. It suits for both short and long summary generation tasks. It observes well the inner connections of the text and expands the analysis by looking further into sentences' voting mechanism.

LexRank is another graph-based approach that differs from TextRank by using a different similarity function: it utilizes an eigenvector centrality in the graphs [19]. In addition to graph based algorithms, sentence extraction has also utilized e.g. hidden Markov models [20] and statistical classification [21].

3.4 Latent semantic analysis

Latent semantic analysis (LSA) is an NLP technique that assumes that the words that have the same meaning occur in similar texts. It finds low-dimensional representations of words and can thus compare them to each other. LSA is widely used in topic modelling.

In LSA the relationships between texts and words are analysed by creating a sparse occurrence matrix where columns represent sentences and rows represent words in the source text. The numbers in the matrix then indicate how many times each word appears in each sentence.

The resulting matrix can make a computationally intractable problem with long text, thus the size of the matrix needs to be reduced. That is done with the help of singular value decomposition (SVD) that can be applied to any $m \times n$ matrix. It strips away some dimensions while preserving the essence of the matrix. SVD for matrix A of size $m \times n$ ($m \geq n$) can be defined as [22]:

$$A = U\Sigma V^T \tag{3.7}$$

where U is an $m \times n$ column-orthonormal matrix whose columns contain the left singular vectors, Σ is an $n \times n$ diagonal matrix who contains the singular values, and V is an $n \times n$ orthonormal matrix whose transpose V^T contain the right singular vectors.

The aim is to find similar rows from the matrix. That is done by comparing the cosines of the angles between two row vectors. The most similar sentences (value close to 1) are treated as the most important sentences and they will form the summary. The angle is measured as following [23]:

$$cos\mu = \sum_{i=1}^{n} ue_i \cdot uf_i \tag{3.8}$$

where U is the first left singular vector of the SVD, ue is the first left singular vector of the summary SVD, and n is a number of unique terms in the full text.

Gong and Liu compared the LSA method in extractive text summarization task with stan-

dard information retrieval method with the aim of finding highly ranked sentences that nonetheless are different from each other and thus cover more content. In the standard method that is done by removing all the words found in the first sentence that was selected in the summary from the remaining the source text and only after that selecting the next sentence. This process is then repeated until the summary has the desired length. [22]

The LSA method takes into account interrelationships between words (e.g. words student, pupil, teacher, and school are related concepts, the first two even synonyms) and can model them near each other in vector space. Once the SVD is performed, the rightmost singular vector is selected from the VT matrix. The sentence with large index value that represents the topic of the vector is then included in the summary. These two steps are performed until the summary has the desired length. As the topic of each singular vector is different, this process ensures that each new selected sentence describes a new salient topic and that way redundancy is avoided.

Gong and Liu found out that the two methods perform equally.

Steinberger and Jezek explain, that the method described previously has some disadvantages that they try to overcome in their enhanced LSA summarizer: the method does not work well if the amount of topics in the text is not known beforehand, and instead of selecting a sentence with the largest index value, it selects a sentence with only a large index value. After they have computed SVD of the matrix whose components are multiplied by corresponding singular values, they calculate the lengths of each sentence vector. The length is also a salience score. Finally the summary is formed of the sentences that have the highest values. [23]

Their enhanced summarizer outperformed other summarizers (Gong and Liu's summarizer and random among others) in all three used evaluation methods: cosine similarity, LSA similarity of the main topic and LSA similarity of the term significance, even though

the differences in the results did not vary a lot in each evaluation.

3.5 Rhetorical structure theory

Rhetorical structure theory (RST) focuses on the text organization, coherence, and relations between text parts and claims that every element in a coherent text has a reason to be included by the author, and thus the text has a fully connected discourse structure that can be presented as a tree. Example of such tree is shown in figure 3.1. [24]

The most frequent structural pattern is the relation between two text parts so that one has a role relative to the other. The part that is more essential to the text is called nucleus and the other part that provides complementary information to the nucleus is called satellite. Some examples of these relations are background, condition, contrast, elaboration, evidence, interpretation, motivation, preparation, and summary. [25]

RST was developed to help in NLG applications [24]. RST-based NLP requires RST parser for the language of the source text. This is the most restrictive factor, as for many languages, e.g. for Finnish, such parser is not known to exist. The Penn Discourse Treebank[3] has more than 40,000 annotated discourse relations for English Wall Street Journal articles and for some of the articles exist human-made summaries. Whole new parser may not need to be developed for each languages as many languages have similar structures. For example Finnish and Hungarian are considered to be discourse-configurational. [26]

Development of new parser is a hard task and would need a lot of annotated training data and gold standard summaries for evaluation. One way to develop a parser is to use a shift-reduce parser and transform the lexical representation of discourse units into a latent space [27]. Further development will most probably be directed towards more advanced

[3]https://www.seas.upenn.edu/ pdtb/

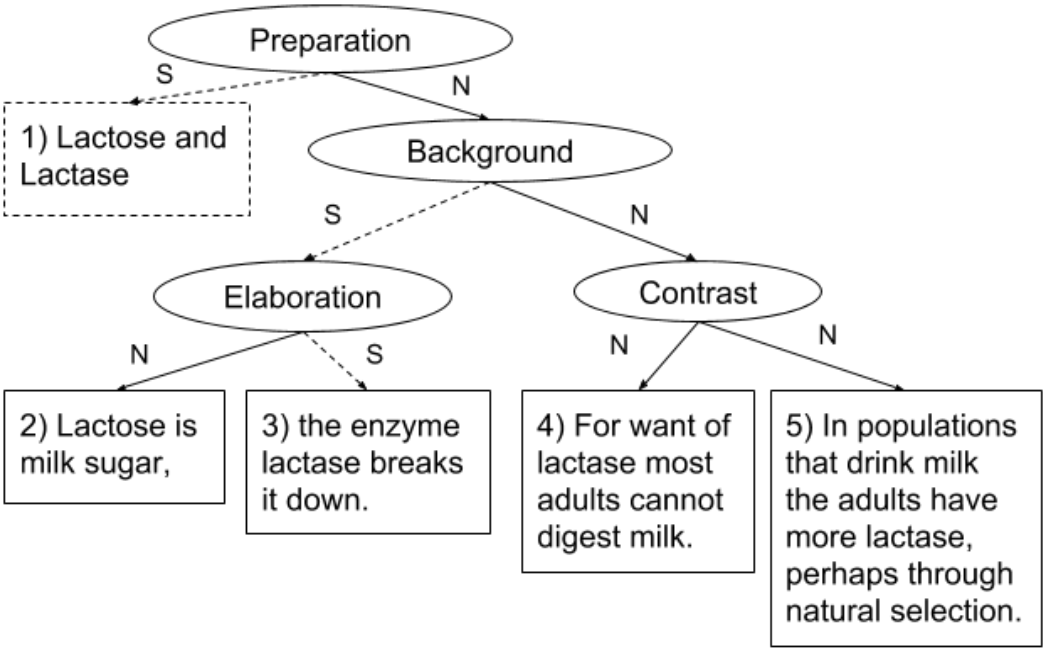

Figure 3.1: RST analysis of a text [24]. S and N represent satellite and nucleus respectively. Contrast is a multinuclear relation: it does not distinguish which text span is more essential.

deep learning.

In summarization task, text is divided in discourse parts, each part is classified based on their importance, unimportant parts are removed, and summary is created of the most important parts. The approach is thus extractive. The goal of this approach is to generate a coherent and well organized summary. There exist a lot of different methods how to rate the parts of text. One of the first ones gives the root of the tree a score that is the number of levels in the tree. Traversing the tree in depth-first mode the score is given to other nodes and decreased by one if a satellite node is found. In another method all the relations have some importance factor and finding satellite node gives the next node a score multiplied with the importance factor. Methods can also utilize e.g. the percentage of nuclei or satellites found in the path from the root to the node and the level of the text part in the tree. These features can also be used to train a machine learning model. Another example of a rule is that a satellite node can only be removed if it is not in the locus effect of the corresponding rhetorical relation. [28]

Louis et al. [29] found that structure information tells more about the importance of a text segment than semantic features. Combining the both is leading to even more improved results. However, equal performance can be achieved with lexical analysis which is much simpler task. Discourse information is still vital when the summary is wanted to be coherent and high quality linguistically so it is expected to be taken into more consideration in future research.

3.6 Neural networks

Current trend in automatic text summarization development is artificial neural networks (ANN). ANNs need a lot of data in the training phase and are then able to handle data that has not been shown before. The methods can be either abstractive or extractive.

Abstractive methods have gained a lot of interest lately as they have potential to generate summaries on human level but extractive methods are still very popular as they are simpler, faster to run and they generate mostly grammatically and semantically correct summaries.

Abstractive summarization

Currently the best methods use sequence-to-sequence models (they map input sequence to output sequence) and long short-term memory (LSTM) model, which is capable of learning long-term dependencies and unlike deep neural networks, it does not have to know the output length beforehand [30, 31]. We start by presenting the base sequence-to-sequence model and then review the latest research that has improved upon it.

The sequence-to-sequence model consist of encoder and decoder. The encoder-decoder architecture is the standard method used in machine translations and in sequence-to-sequence prediction, which summarization task is. Usually recurrent neural networks (RNN) are used for both encoder and decoder. RNNs analyze time series data, i.e. sequences or arbitrary lengths, and predict the future. Thus they are great in NLP tasks where text is a sequence. What distinguish them from simpler feedforward neural networks is their ability to memorize previous steps as the state is updated after each output is formed, but when the amount of steps grows, RNNs capability of connecting the information decreases. [32]

LSTM is a kind of RNN and helps in the memorizing problem being able to remember information for long periods of time. Basic RNN has a single tanh layer in the hidden state but LSTM has four neural network layers: one tanh layer and three sigmoid layers. That is represented in the figure 3.2. On top of the figure the horizontal line represents the cell state Ct that is updated through gates. A gate consists of a sigmoid neural net layer (marked with) and a pointwise multiplication operation. A sigmoid layer (called

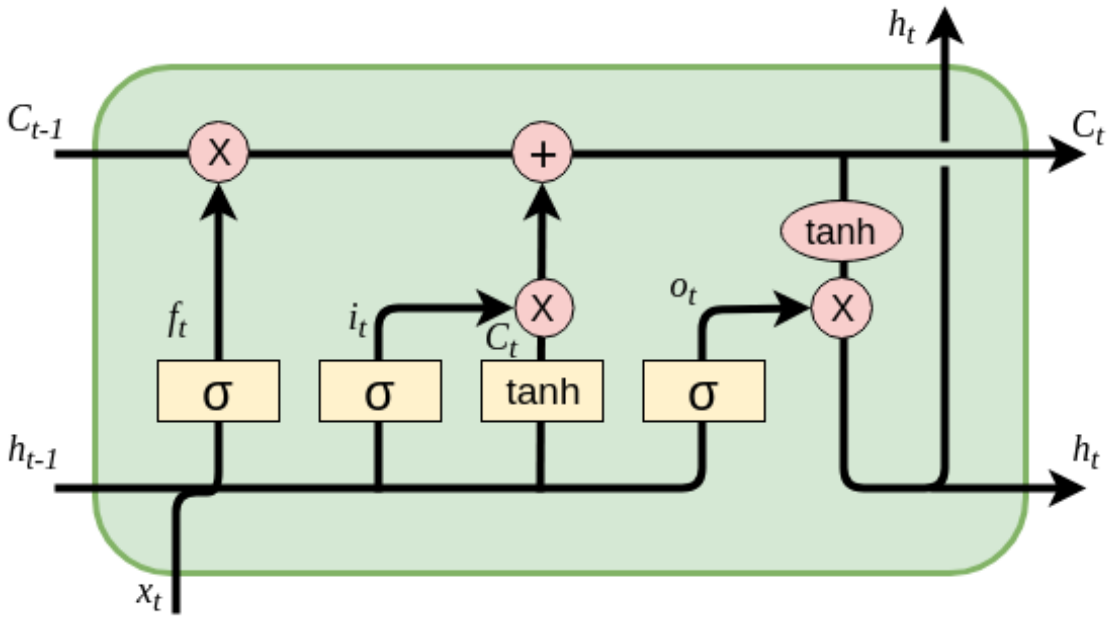

Figure 3.2: LSTM model. Pink circles represent pointwise operations like addition and multiplication, and yellow boxes neural network layers. Each line carries a vector and merging represents concatenation and forking copying. [33]

forget gate layer) outputs a number between 0 and 1 telling how much information should be forgotten. The next step decides what new information will be stored in the state: a sigmoid layer (called input gate layer) decides which values will be updated and tanh layer creates a new vector of new candidate values that will be added in the state. Last step is to update the cell state and decide what to output. [33]

In a basic encoder-decoder flow, the encoder first reads an input text word by word or phrase by phrase, where end-of-sequence token is added to the end, and transforms it to a distributed representation. In the distributed representation a concept is represented with more than one neuron and one neuron represents more than one concept. It is thus dense and different from sparse representation that needs a new dimensionality each time a new concept needs to be included. Using a multi-layer neural network the distributed representation is combined with the hidden layers that were generated when the previous

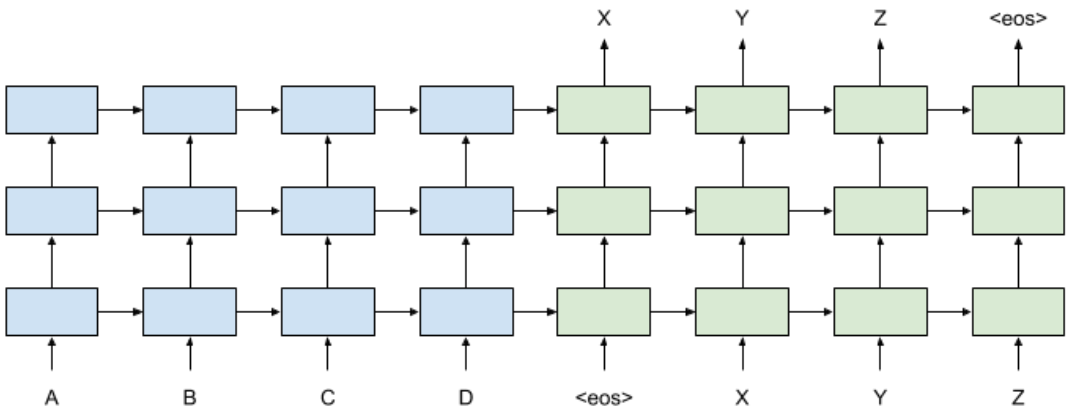

Figure 3.3: Encoder-decoder neural network architecture where an input sequence ABCD is converted in the blue encoder into a target sequence XYZ in the green decoder. ¡eos¿ is the placeholder for end-of-sentence.

word was processed. [34]

The decoder then processes the distributed representation after the last word of the text input has been encoded. Then it utilizes a softmax layer and an attention mechanism to generate the summary of the input text. Each freshly generated word is given as an input when generating the next word. [34] Encoder-decoder architecture is presented in the figure 3.3.

In his research, Lopyrev [34] used four hidden layers in his LSTM network, each having 600 hidden units. He also compared two different attention mechanisms that compute weight to each input word to determine how much attention should be paid to it. He trained the model to generate headline for English news articles keeping only the first paragraph of the input text. Only 40,000 most frequent words were kept and too long headlines or texts were filtered out. Division to training and tests sets were done based on the time of an article so that articles published near each other do not appear in both sets. The problems in the used data set included headline not summarizing the article very well or containing something irrelevant. However, bad articles were not removed out as the

ANN should be able to handle them.

There is a problem with using fixed size vocabulary: the performance degrades if the output summary needs words that are not included in the vocabulary. This is a bigger problem in languages that have a rich vocabulary, e.g. Finnish. Some solutions have been introduced to fix this problem, but Jean et al. describe a way to increase the vocabulary size instead, without increasing computational complexity too much. In the training phase, he partitions the training corpus and defines a small subset of the target vocabulary to be used for training each partition. [35]

Nallapati et al. continued the development by using bidirectional GRU-RNN in the encoder and a unidirectional GRU-RNN in the decoder, an attention mechanism and a soft-max layer. They introduced several ways to still decrease the vocabulary size, e.g. by adapting Jean et al. [35] they restrict the words in decoder to the words in the partition's source documents, and by setting the vocabulary size to be a fixed size of most frequent words. That helps to reduce the size of the soft-max layer of the decoder and thus speeds up the process. They also introduce the use of a sigmoid activation function to handle out-of-vocabulary use: the decoder can generate a pointer to a word in the source text that is then copied into the summary in a case when the word does not appear in the training vocabulary. [36]

Improved pointing method has also been used in other research [37] where also the problem with repeating content has been addressed: coverage vectors keep track on the coverage of words in the source document. The vector is the sum of all attention distributions. When the attention mechanism decides where to attend next, it will avoid old locations and thus repetition.

Current state-of-the-art method, called ATSDL, combines the LSTM model with convolutional neural network (CNN) [31]. First keyphrases are extracted from the source text and divided into subject phrases, relations phrases and object phrases. E.g. "I (subject

phrase) travel to (relational phrase) Finland (object phrase)". Phrases with similar semantics (if one is a part of another or in case of hypernym[4]) are combined together to avoid redundancy. The phrases are then used to train the LSTM-CNN model. The summary generation step is divided in two parts: there is a threshold for conditional probability that defines whether generate or copy mode is used. In generate mode the next phrase is predicted normally. In copy mode a generated phrase is not believed to form a coherent summary, so the previous phrase location information is used to copy the following phrase. This arrangement helps to deal with rare words and formulate higher quality summary.

The benefits of the sequence-to-sequence approach are the small memory that is needed, it works without any language catalogs and it does not need extensive domain knowledge. Training time on the other hand can be long. Soft-max layer in the decoder is computationally most expensive part of the architecture [36]. After training, the model is fast to generate summaries.

As the LSTM approach with encode-decoder works well in English generating grammatically correct summaries most of the time [34] and does not need any language catalogs, it should work well in any language. It is more of a question where to find large enough data set to train the network. The type of text also matters: the data set used in [34] contained only news articles and did not work well to summarize other type of text as the structure differs. To make an universal summarizer the model would need to be trained with all kinds of text. Human-made summaries can also be too far reach in current abstractive development: they cannot be formed only from source text and they are more abstractive than automatically generated summaries [38]. On the other hand, human-made summaries are found to follow some common latent structures, such as "who action what", that could be integrated into the algorithm and the quality would improve [39]. Abstractive summa-

[4]a word with a broad meaning constituting a category into which words with more specific meanings fall

rization has thus multiple possible paths for future development.

In machine translation applications with a similar architecture, it was often noted that the performance is the worse the lengthier the source sentence is [40] but by reversing the order of words in the input sentence Sutskever et al. [30] were able to get rid of the problem.

Abstractive summarization has also been used to generate Wikipedia articles from several source documents [41]. Extraction to minimize the size of the input was combined with abstraction to generate the Wikipedia article. In the abstraction phase a decoder only sequence transduction model was used. It is better than traditional encoder-decoder architecture on longer texts.

Even though produced summaries are often literally similar or related to each other, they do not always be semantically similar, i.e hold the same meaning. E.g. bus and motorcycle are related and relatively close to each other in a vector representation, but replacing them with each other in the text would change the meaning. A Semantic Relevance Based neural network model (SRB) works otherwise similarly as the encoder-decoder described above but has a similarity component measuring the relevance of source and generated texts. In the training phase the similarity score is maximized to achieve high semantic relevance. The used function is cosine similarity that measures the distance between two vectors. [42]

Extractive summarization

The best methods in extractive summarization do not depend on human annotated data because it is hard to get, but are data-driven and figure out the features by themselves with neural networks.

The state-of-the-art method in extractive summarization is a RNN based sequence classi-

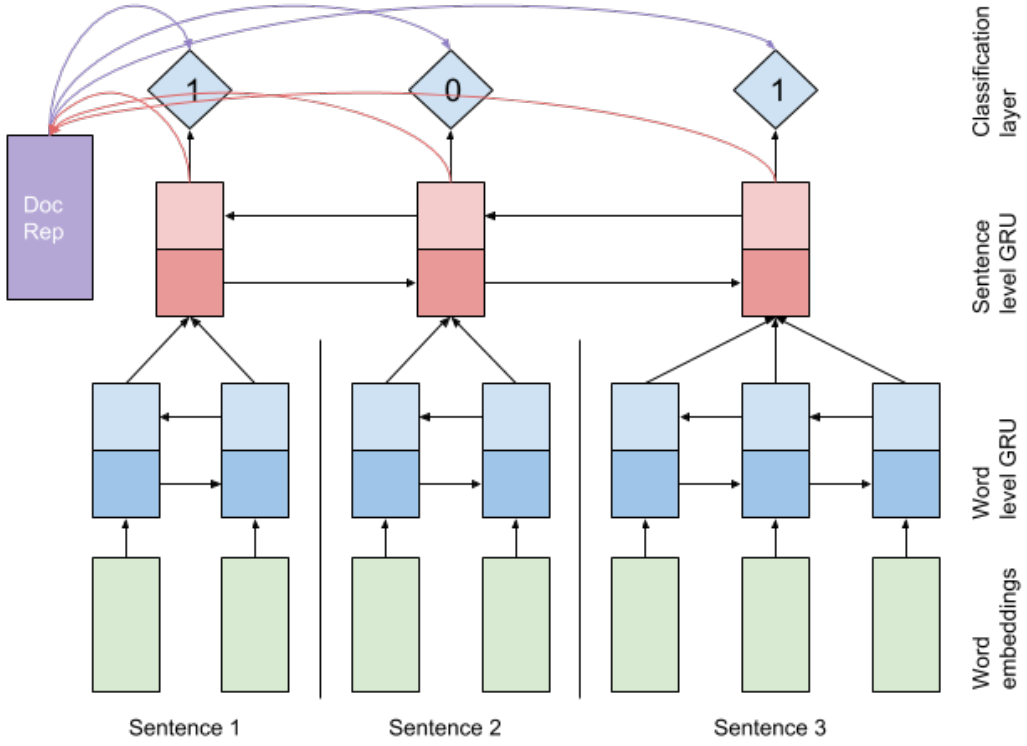

Figure 3.4: A two-layer RNN based sequence classifier [43]

fier by Nallapati et al. [43] called SummaRuNNer. The method goes through sentences in the source text one by one and decides whether to keep them in the summary or not. It uses a two-layer bi-directional GRU-RNN with update and reset gates. In the first layer a RNN works at the word level and computes hidden state representations for each word from the first to the last word. Another RNN at the word level runs backwards from the last word to the first. On the second layer there is also a bi-directional RNN that runs at the sentence level and takes as an input the hidden states of the word level RNNs. The sentence level hidden states are then formed into a representation of the whole document. On the top layer is a sigmoid activation based classification layer that makes the binary classification whether sentence belongs to a summary or not. The whole architecture is presented in figure 3.4.

In SummaRuNNer, good summary sentences are defined as rich in terms of content, providing something new in the summary and being salient in the source text. In order to

train the model, they changed the gold standard abstractive summaries to needed sentence labels with unsupervised approach where the goal is to select sentences that maximize the ROUGE score. Instead of testing with all combinations, they approach the problem in a greedy way by adding sentences to the summary as long as the ROUGE score increases. The sentences that are selected this way will then form the training data set. [43]

The question how long the summary should be can be addressed in several ways. Cheng and Lapata rerank the selected sentences based on the probability scores from the softmax layer and select three top sentences [44]. Nallapati et al. suggest dynamically setting the summary length based on the probability distribution [43]. It is also stated that indicative summary (only states the main idea for reader to decide if it is worthwhile to read the whole text) can be as short as two sentences, but the length of informative summary should not be restricted [20].

Chapter 4

Problem definition and the data

In this chapter we will discuss about our research questions, and introduce the data that was used in the automatic text summarizer.

4.1 Problem definition

As discussed in chapter 2, there are endless applications and use cases for summarization tools, and the methods used in those tools can also be utilised in various other NLP and AI tasks. In this thesis, we want to test how well the extractive summarizers work with Finnish news text, as so little research has been done with Finnish summarizers in the recent years. We are interested in seeing if there are any surprising things that need to be taken into account with Finnish texts or news texts. Another interesting aspect is the quality of the data: would it suit well in training a ANN based summarizer in the future?

The goal is to build an unsupervised news summarization tool that reduces the article size. As will be explain in the next subchapter, we have a title, summary and text for each news article and it is interesting to see how well we can extract the title and summary from the

text.

4.2 The data

The data set used in this thesis contains approximately 14,000 Finnish news articles published in the internet from August 2017 to January 2018. Each article contains a text, a summary of one or two sentences, a title, a publishing date, and other metadata. The summary is considered to be a gold summary. The text content is initially in HTML format but turned into plain text in the preprocessing phase.

The top 20 tags are (occurrences in brackets):

1. Domestic news (5959)
2. Foreign countries (4086)
3. Finland (1975)
4. Politics (1440)
5. Economy (1053)
6. Helsinki (841)
7. Culture (781)
8. Finnish politics (679)
9. Nordic countries (664)
10. Lapland (638)
11. Finnish politicians (629)
12. Police (595)
13. Finnish political parties (566)
14. Russia (552)
15. Political parties (534)
16. Transport and transportation (445)
17. Middle East (429)
18. European Union (425)
19. Children (family members) (424)
20. Tampere (417)

The news articles contain over 5,000,000 words out of which the amount of words considered as stopwords is over 900,000. The amount of different stems (excluding stop words) is over 200,000. Figure 4.1 shows the distribution of top 15 of those stems.

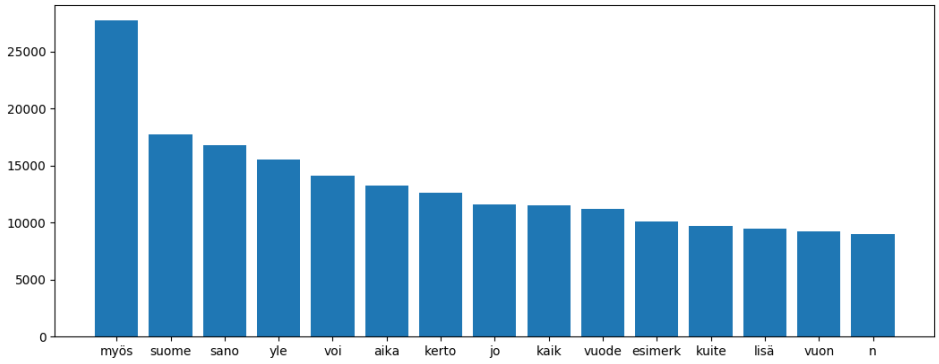

Figure 4.1: Top 15 stem words (stopwords not included)

In the beginning, we remove from the data set articles that for some reason do not have either text, summary or title. There are almost hundred such articles in the original data set.

On average, the length of the news text is 30 sentences. The amount of sentences in the summary is on average 9% of the sentences in the original text. Equivalent number for the amount of words is 8%. Still the article length varies a lot: shortest articles have couple of sentences whereas longest ones have even 300 sentences.

As we noted in section 3.5, in RST research the importance of structure information is significant. In news articles the most important sentence is usually in the beginning and that can be seen also in our dataset. The most important sentence contains the key idea that is then elaborated in the following sentences. Even though the location information seems a good feature to use, we decide not to use it as it is very much data specific feature, and for news data from some other source the place of the key sentence would need to be found again.

Chapter 5

Method

This chapter describes the steps that are taken to develop the summarizer for Finnish news data. In the development we follow CRISP-DM (Cross-Industry Standard Process for Data Mining) [45], that is presented in figure 5.1.

The purpose of the study and the data have already been reviewed in the chapter 3. Next we will go through the steps taken in preparing the data and developing the models. Finally we will evaluate the results and see the summarizer in action. The feedback loop in the figure 5.1 shows that findings in the later steps can and should trigger corrections in the previous steps.

5.1 Preprocessing

One of the biggest challenges in analysing textual data is how to change the unstructured data to a structured format in the preprocessing phase. It is a crucial step and needs to be done before the actual analysis.

For the data used in this thesis, the preprocessing consisted of the following steps: dehtmlifying, case normalization, tokenization, stopword removal, and stemming.

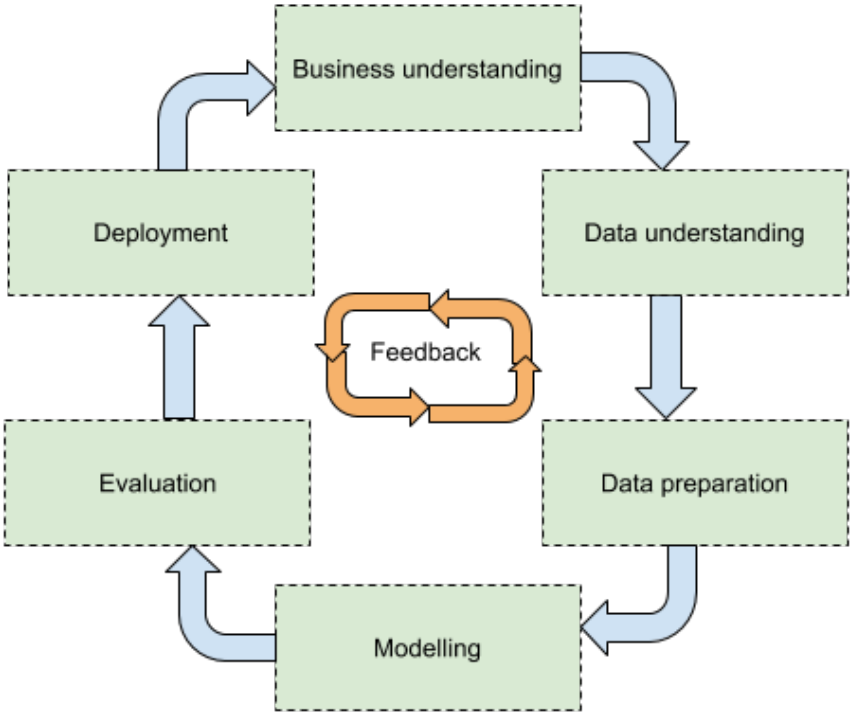

Figure 5.1: CRISP-DM process flow for text mining

The data exist in HTML format so the first step in the preprocessing is to turn it into plain text and get rid of HTML tags. Before this can be done, any inconsistencies (e.g. missing closing tags) in HTML are fixed manually. In this phase we also remove articles that miss either text, summary or title.

Tokenization means splitting text into words and removing punctuation. In Finnish the way to do it is to use white space and punctuation to delimit word boundaries. However, this may not result with the desired outcome if the word is an abbreviation such as i.e. (abbreviation for in other words) because the punctuation splits the letters into separate tokens. The text has some hashtags from which we remove the hash character #.

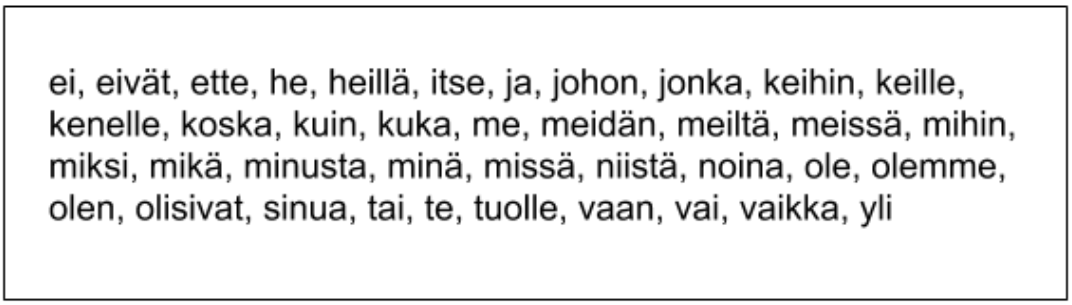

Figure 5.2: Some Finnish stopwords

In stopword removal (stopping) frequently occurring uninformative words are removed to speed up the processing. Figure 5.2 shows some examples of Finnish stopwords. Depending on the data there may be other words that also should be removed, but in this case we trust an existing list of stopwords.

Stemming strips prefixes and suffixes from words and turns them into their stem forms. After that for example *run* and *running* are considered as the same stem *run*, and probably some spelling mistakes are cleared from the data. Stemming reduces dimensions in word vector space and improves the algorithms.

Finally we convert all the characters to lowercase. In Finnish language, capitalization

helps human to recognize proper nouns. Capitalization is also used in the first word of the sentence but that word should be treated the same way as every other word in a sentence, thus we turn all the characters into lowercase.

5.2 Modelling

Our experiments are structured as follows: given the original news text, we generate a couple of sentences long summary and compare it to the given summary. Each summary will be generated in five ways: randomly, selecting longest sentences, with TextRank algorithm, with intersection method, and with frequent words method. At the end they are evaluated with ROUGE-1 method.

Randomly generated summary gives us a baseline of what could be achieved without any knowledge. It is formed by randomly picking sentences from the source text until the desired length is achieved. ROUGE scores are averaged across 30 runs as the process is stochastic.

Longest sentences form the summary of the longest sentences and is thus a good benchmark for other methods. If other methods exceed its performance, we know that they do not prefer longer sentences, which is a good thing.

The intersection method calculates a score for each sentence based on common words with other sentences. The intuition is, that a sentence that has many similar words with other sentences cover the most content and is thus a good summary sentence. To avoid preference towards long sentences we normalize the score by dividing it with average number of words in the sentences.

The frequent words method on the other hand prefers sentences that have the most frequent words. It is similar with the intersection method but rates the sentences a bit differently. It also clearly prefers longer sentences if they just have many frequent words.

To perform the TextRank algorithm we utilize the open source *summa*[1] library.

In the actual summary generation, the desired length needs to be defined beforehand. The lengths of the news articles vary in the data set so it is hard to set a constant for the amount of sentences that the summary should be so that it would work well in every case. In the source documents the length of the summary (in terms of words) is approximately 10% of the original news article text, thus we think it is appropriate to use it. Adding more sentences to the summary is interrupted when the word count is exceeded.

The automatic text summarizer is written in Python 3 programming language. We utilize the nltk[2] library in natural language processing tasks.

5.3 Results and evaluation

The summarizer is evaluated with the ROUGE-1 evaluation method. Random summaries and summaries with the longest sentences of the original text are also generated so that the evaluation scores can be compared to something that is generated with the same data but in a stochastic way.

Precision is a bad measure in this case as the length of the generated summary is defined based on the length of the initial text, and not based on the content coverage. Thus the precision score gets worse each time something that is not in the golden summary is included in the generated summary. Table 5.1 shows the scores for each summarization method.

Table 5.1: Evaluation results

	Random	Longest	Frequent	Intersection	TextRank
ROUGE-1	0.10	0.10	0.15	0.15	0.12

[1]https://pypi.org/project/summa/

[2]https://www.nltk.org/

When reading the generated summaries and comparing them to the original summaries, some summaries are unexpectedly good. Table 5.2 presents one of the good ones generated with the frequent method. The range of ROUGE-1 scores is wide: many of the summaries get a score of 0 because the golden summary contains words that have not been seen in the text. Respectively sometimes the scores are as high as 0.8. On average they lie below 0.20.

On the other hand, sometimes the generated summaries do not correspond the original summary. The problem is not necessarily in the summarizer, but in the original summary. Example can be seen in a table 5.3. It shows two problems in the data set: the original summary does not convey the key points of the article but only tries to raise the interest of the reader to open and read the whole text and even the sentences in the same text do not tell the same story. This raises a question whether the summaries in the data set can be considered as gold summaries and would they be good in training a ANN model.

Against gold standard summary *”Hurrikaani Maria on nyt neljännen luokan hurrikaani” / ”Hurricane Maria is now a fourth category hurricane”* one of the generated summaries *”Karibianmerellä voimistuva hurrikaani Maria on saavuttanut neljännen luokan hurrikaaniasteikolla.” / ”Maria Hurricane has intensified in the Caribbean Sea and reached the fourth category on the hurricane scale”* got the ROUGE-1 score of 0.62. These are individual examples of the problematics of individual measures. The generated summary has the same information and more than the gold standard and thus could be interpretated as perfect.

Table 5.2: Example of a good summary generated with the frequent method.

Original summary
Suomesta on tarkoitus tehdä avaruustoiminnan huippumaa vuoteen 2020 mennessä. Tätä tavoitetta varten pitäisi maahamme perustaa oma avaruushallinto ja avaruusministeri, sanoo Aalto-yliopiston professori.
Finland is expected to be made the top country of space activities by 2020. Our own space management and space ministry should be be set up for this goal, says a Professor of Aalto University.
Frequent
Suomeen suunnitteilla oma avaruushallinto. Tavoitteeseen pääsemiseksi Suomeen on suunnitteilla oma kansallinen avaruuslaki. Euroopan avaruusjärjestö ESA ilmoitti perustavansa Suomeen yrityskiihdyttämön, jonka tavoitteena on saada Suomeen avaruusalan kasvuyrityksiä sekä mahdollistaa teollisuuden yhteys uuden avaruusteknologian sovelluksiin.
Finland has planned its own space management. To reach the goal a national space law is planned in Finland. European Space Agency ESA announced that it would set up a business accelerator in Finland. The purpose is to enable the growth companies in the space industry to make the connection of industrial manufacturing to new space technology applications possible.
Random
Avaruudessa on jo satoja noin parin kilon painoisia satelliitteja. Nanosatelliittien kysyntä on kasvanut räjähdysmäisesti. Tällä hetkellä avaruuden valloitusta tehdään erityisesti kevyiden, niin kutsuttujen nanosatelliittien avulla, joiden kysyntä on kasvanut viime vuosien aikana räjähdysmäisesti.
There are already hundreds of satellites weighing a few kilograms in the space. Demand for nanosatellite has expanded explosively. Today the conquest of space is made with nanosatellites, whose demand has grown explosively in the recent years.

Table 5.3: Example that shows the poor quality of original summaries in some of the articles in the data set.

Original summary
Trump vastasi toimittajien kysymyksiin tavatessaan Norjan pääministerin Washingtonissa.
Trump responded to journalists' questions when meeting with Norwegian Prime Minister in Washington.
Frequent
Trump sanoi, että on mahdollista, että Yhdysvallat palaa Pariisin ilmastosopimukseen, jos sen ehdot neuvotellaan Yhdysvaltojen kannalta paremmiksi.
Trump also said that it is possible that the United States will return to the Paris Climate Treaty if its terms are negotiated better for the United States.
Random/Longest
Erikoistutkija Muellerin lähipiiri on viime aikoina ilmaissut kiinnostuksensa haastatella Trumpia, ja Trumpin asianajajat ovat kertoneet, että he tulevat suostumaan Muellerin pyyntöihin.
Mueller's close associate recently has expressed interest in interviewing Trump. Trump's own lawyers have said they will agree to Mueller's requests.
TextRank/Intersection
Yhdysvaltain presidentti Donald Trump sanoo, ettei todennäköisesti suostu vaalikampanjansa ja Venäjän kytköksiä tutkivan Muellerin haastatteluun.
US President Donald Trump says he is unlikely to agree to an interview with Robert Mueller, his campaigner and Russian interlocutor.

Chapter 6

Discussion

In the last chapter we presented a comparative research on different summarizing algorithms on Finnish news text. The ROUGE-1 measures did not indicate good performance as on average they were below 0.20, but qualitative analysis shows that many of the summaries sound good and even better than the original gold-standard summary.

The used methods work well in Finnish news text. No dependency parsing or PoS tagging is needed which makes the method language independent. The only thing that would possibly need to be changed for other languages is the preprocessing step where some language or data specific things (e.g. stopwords, tokenization) needs to be considered. If language detection was added, the algorithms would automatically work with almost any language.

Finnish news data is high quality text. Examples in tables 5.2 and 5.3 show how sentences are complete, they do not contain too much references to other sentences but work well alone. That makes the data very suitable for extractive summarization where sentences may be taken away from their context.

It is arguable though, if the news data available today is good for quality summary generation. It is a trend that the purpose of the title and the abstract is to raise an interest in

the reader to read more and open the article. If the key point is revealed already in the summary, the willingness to open the article decreases. This would explain why the so called gold standard summary does not contain same words that the actual text repeats. All the three purposeful algorithms play with the same question: which words exist in the text and how often, and do not care about synonyms.

What is good, is that the longest and the random algorithms perform a little bit worse than the rest. While longer sentences may catch more frequent words, they also catch more words that do not exist in the gold summary and that affects the ROUGE score negatively.

It seems that we are trying to solve a different problem than what the data is capable of for. The summaries in the data set do not try to convey the key idea of the text. The data set would need human annotations: key sentences should be marked and generated summaries be evaluated against them. Also, the hurricane Maria example at the end of the chapter 5 shows how problematic the ROUGE-1 metric is: in an application where the goal is to make news easier to read for humans, couple of additional words may bring more value even if they decrease the evaluation score! 0.62 is a good score, but for human eyes the summaries seem equally good.

What can be seen is that relatively long gold summary decreases the ROUGE-1 scores. The shorter the gold summary and the longer the generated summary, the more probable it is to get a high score.

There are many ways to improve upon the algorithms used in this thesis: one direction is the improvement of the data set that we already mentioned. Other way is to calculate the vector distances of words and take into account synonyms and words that are close to each other. That would need a lot of quality data. With the help of PoS information only e.g. nouns and adjectives could be considered and verbs that in the news context seem to be more varied could be ignored. Even though ROUGE scores are the standard metric in the

field of automatic summarization, more relevant metrics could also be thought of.

Chapter 7

Conclusion

In this thesis, our goal is to present the essence of automatic text summarization. After short introduction to the topic, chapter 2 presents the history and applications of automatic summarization. In chapter 3 we introduce us to the various methods with with summarization tasks can be approached. It covers the naive way that is build upon simple heuristics, a bit more advanced graph-based models, latent semantic analysis, rhetorical structure theory, and neural networks. We see how the methods have evolved and get to know their drawbacks and advantages. Even though the current trend is in neural network based development and lot of research is done in the area of abstractive summarization, extractive summarization is still a popular research area due to its simple nature and quickly run software. In many applications the results that can be accomplished with extractive methods are enough, and to achieve excellent results with abstractive ways needs a lot of progress in the areas of NLU and NLG. What comes to language support, the methods introduced in this thesis are language independent and should work with any language with only some changes done to the preprocessing step. The most restrictive factor in machine learning approaches is the availability of proper training data.

In chapter 4 and forward we introduce our own summarizer that is built to decrease the

size of Finnish news data. We get hands-on experience on how the language affects the task and what to take into consideration when developing a NLP system with other language than English. Such a quality sentences we have in our Finnish news data set are an advantage in extractive summarization: sentences easily fit together and no information is lost in missing references. On the other hand we notice that the gold summaries in the data may not be a good benchmark for the generated summaries, as their purpose is more to raise interest towards the article than to tell the key point. That can be seen in the ROUGE measures that are not as good as could be expected when manually reviewing the summaries.

As NLP and related fields make progress, new possibilities for text summarization unfold. However, still many challenges are waiting to be solved. The biggest challenge is the availability of data especially outside news domain as every new application would need proper data in their own domain and in the specific languages. In this thesis we notice that different kinds of texts have different structures and the most important content may lie in different parts of the text. Evaluation also needs high quality gold-standard summaries which are laborious to create manually.

In addition to the ROUGE evaluation measure that evaluates information coverage, other measures are needed to evaluate coherence, semantic similarity, and overall quality of the generated summary. To improve the quality, ways to integrate discourse parsing and RST as well as semantic parsing need to be developed further.

References

[1] Gary Miner, John Elder IV, and Thomas Hill. *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press, 2012.

[2] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2):159–165, 1958.

[3] Harold P Edmundson. New methods in automatic extracting. *Journal of the ACM (JACM)*, 16(2):264–285, 1969.

[4] Megan Squire. *Mastering Data Mining with Python–Find patterns hidden in your data*. Packt Publishing Ltd, 2016.

[5] Shamima Mithun and Leila Kosseim. Summarizing blog entries versus news texts. In *Proceedings of the Workshop on Events in Emerging Text Types*, pages 1–8. Association for Computational Linguistics, 2009.

[6] Minqing Hu and Bing Liu. Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760, 2004.

[7] Roshna Chettri and Udit Kr Chakraborty. Automatic text summarization. *International Journal of Computer Applications*, 161(1), 2017.

[8] 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 Lan-*

guage Technology-Volume 1, pages 71–78. Association for Computational Linguistics, 2003.

[9] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[10] Hans Moen, Juho Heimonen, Laura-Maria Murtola, Antti Airola, Tapio Pahikkala, Virpi Terävä, Riitta Danielsson-Ojala, Tapio Salakoski, and Sanna Salanterä. On evaluation of automatically generated clinical discharge summaries. In *PAHI*, pages 101–114, 2014.

[11] Richard Braddock. The frequency and placement of topic sentences in expository prose. *Research in the Teaching of English*, 8(3):287–302, 1974.

[12] Craig G Smith. Braddock revisited: The frequency and placement of topic sentences in academic writing. *The Reading Matrix*, 8(1), 2008.

[13] Fred Karlsson. *Yleinen kielitiede*. Yliopistopaino/Helsinki University Press, 2008.

[14] Erkka Leppänen. Homografiongelma tekstihaussa ja homografien disambiguoinnin vaikutukset. *Informaatiotutkimus*, pages 133–144, 1996.

[15] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1262–1273, 2014.

[16] Hassan H Alrehamy and Coral Walker. Semcluster: Unsupervised automatic keyphrase extraction using affinity propagation. In *UK Workshop on Computational Intelligence*, pages 222–235. Springer, 2017.

[17] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[18] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998.

[19] Günes Erkan and Dragomir R Radev. Lexpagerank: Prestige in multi-document text summarization. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, 2004.

[20] John M Conroy and Dianne P O'leary. Text summarization via hidden markov models. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 406–407. ACM, 2001.

[21] Julian Kupiec, Jan Pedersen, and Francine Chen. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 68–73. ACM, 1995.

[22] Yihong Gong and Xin Liu. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 19–25. ACM, 2001.

[23] Josef Steinberger and Karel Jezek. Using latent semantic analysis in text summarization and summary evaluation. *Proc. ISIM*, 4:93–100, 2004.

[24] Intro to rst. https://www.sfu.ca/rst/01intro/intro.html. Accessed: 2018-04.

[25] Mann William and Sandra Thompson. Rhetorical structure theory: towards a functional theory of text organization. *Text*, 8(3):243–281, 1988.

[26] Jorunn Hetland and Valéria Molnár. *Structures of focus and grammatical relations*, volume 477. Walter de Gruyter, 2003.

[27] Yangfeng Ji and Jacob Eisenstein. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 13–24, 2014.

[28] Vinícius Rodrigues Uzêda, Thiago Alexandre Salgueiro Pardo, and Maria Das Graças Volpe Nunes. A comprehensive comparative evaluation of rst-based summarization methods. *ACM Transactions on Speech and Language Processing (TSLP)*, 6(4):4, 2010.

[29] Annie Louis, Aravind Joshi, and Ani Nenkova. Discourse indicators for content selection in summarization. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 147–156. Association for Computational Linguistics, 2010.

[30] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[31] Shengli Song, Haitao Huang, and Tongxiao Ruan. Abstractive text summarization using lstm-cnn based deep learning. *Multimedia Tools and Applications*, pages 1–19, 2018.

[32] Aurélien Géron. *Neural networks and deep learning*. O'reilly Media, Inc, 2018.

[33] Understanding lstm networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs, 2015. Accessed: 2018-04.

[34] Konstantin Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*, 2015.

[35] Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. On using very large target vocabulary for neural machine translation. *arXiv preprint arXiv:1412.2007*, 2014.

[36] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016.

[37] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

[38] Jackie Chi Kit Cheung and Gerald Penn. Towards robust abstractive multi-document summarization: A caseframe analysis of centrality and domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1233–1242, 2013.

[39] Piji Li, Wai Lam, Lidong Bing, and Zihao Wang. Deep recurrent generative decoder for abstractive text summarization. *arXiv preprint arXiv:1708.00625*, 2017.

[40] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

[41] Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*, 2018.

[42] Shuming Ma and Xu Sun. A semantic relevance based neural network for text summarization and text simplification. *arXiv preprint arXiv:1710.02318*, 2017.

[43] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*, pages 3075–3081, 2017.

[44] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[45] Pete Chapman. Crisp-dm 1.0 step-by-step data mining guide. 2004.