Keyphrase Extraction: an Ensemble Approach

Luca De Vita

A thesis submitted in fulfillment
of the requirements for the degree
of Master of Science in Business
Analytics

Keyphrase Extraction: an Ensemble Approach

Extracting keyphrases using unsupervised ensemble methods and graph databases

Luca De Vita

Vrije Universiteit Amsterdam
Faculty of Science
Business Analytics
De Boelelaan 1081a
1081 HV Amsterdam

Host organization:
BearingPoint B.V.
De Entree 89
1101 BH Amsterdam

July 30, 2020

Preface

This thesis has been written to fulfil the requirements for the Master Project Business Analytics at the Vrije Universiteit Amsterdam. This graduation thesis consists of a six months internship, which was conducted from February 2020 till July 2020 for the amount of 36 ECTS. During this internship, the goal is to combine academic research with real-world problems.

I would like to thank my supervisor, Dr. F. Kunneman of the Vrije Universiteit for his guidance and his detailed feedback on draft versions of this thesis. Furthermore, I would like to thank M.Sc. N. Meibergen for fulfilling the role of external supervisor at BearingPoint. N. Meibergen introduced me to graph databases and good software engineering practices, which both proved useful during this thesis. Moreover, I would like to thank Prof. Dr. S. Bhulai for being the second reader, and his useful feedback in the initial stages of this research.

Lastly, I would like to thank both BearingPoint and Vilans for the opportunity to conduct this research. Specifically, I would like to thank the employees of Vilans for their time to annotate the documents. Without human annotations it would have been infeasible to evaluate the developed methods, for this specific domain.

Executive summary

Problem definition: Businesses are increasingly interested in using natural language processing techniques to extract relevant information from the plethora of textual data. An important subfield in natural language processing is called keyphrase extraction. Keyphrases can concisely summarize the content of a document, and have a wide variety of purposes. More specifically, they can improve search engines and can aid the performance of many other natural language processing tasks such as text categorization. However, many current state-of-the-art techniques provide unsatisfactory results. Therefore, new approaches are required to advance the field.

Academic/Practical Relevance: This research focuses on three major areas. Firstly, there is a lack of consensus among researchers regarding good agreement measures for keyphrase extracion. For this reason, we propose two appropriate agreement measures that are used in related natural language processing tasks. Secondly, we show that combining statistical and graph-based algorithms into one model (hereafter ensemble) can greatly improve the results for a wide variety of evaluation metrics. Lastly, we propose a graph-based approach to deal with one of the main challenges in this field: extracting keyphrases that summarize the text, but are *not* present in the text.

Methodology: The agreement measures were used to determine the granularity of the annotated keyphrases. Thereafter, we implemented several existing unsupervised algorithms, but combined these approaches in a post-processing phase to create two ensemble methods. Lastly, graph databases and a domain-specific thesaurus were used to enrich these keyphrases.

Results: By comparing the keyphrases of the algorithms with the keyphrases from three domain-experts, we observed that our ensemble methods outperformed current state-of-the-art algorithms. Specifically, in comparison with the best performing individual unsupervised algorithm the best performing ensemble obtained a 14.7% relative improvement, for the mean average precision of the top five ranked keyphrases. Moreover, our experimental results suggest that the ensemble methods are robust for short and long documents. Additionally, we provide anecdotal evidence that graph databases can enrich the final set of keyphrases, by adding domain-specific information related to the document.

Managerial implications: Implementation of our method could improve document retrieval in information systems, which may lead to an increase in employee productivity.

Contents

Abbreviations

AP average precision. 47

DNNs deep neural networks. 11–14, 17, 67

IAA inter-annotator agreement. 3, 8, 19–22, 34, 37–39, 45, 46

MAP mean average precision. 3, 47, 49, 50, 52, 55, 56, 58, 59, 64, 67

NLP natural language processing. 2, 6, 9, 12, 16, 21, 38, 45, 50, 64

tf term frequency. 24, 55, 56, 58, 64

tf-idf term frequency-inverse document frequency. 11, 14, 16, 18, 24–26, 55, 56, 58, 63, 65

Chapter 1

Introduction

During the last two decades applied researchers have become increasingly interested in automatic[1] keyphrase extraction. The goal of keyphrase extraction is to find phrases from a document which are most related to the main topic(s).[2] This vast amount of research opens up opportunities for companies to use keyphrase extraction for a wide variety of purposes. More specifically, keyphrases can provide a concise summary of a document, and can be used as an index for information retrieval purposes (Nguyen and Kan, 2007; Turney, 2000). Furthermore, they can be useful for a wide variety of other natural language processing (NLP) tasks such as text classification (Hulth and Megyesi, 2006).

Most studies have mainly focused on algorithmic development, yet the consensus among researchers is that state-of-the-art performance is lower than many other NLP tasks (Hasan and Ng, 2014). Moreover, there is a lack of consensus regarding the effectiveness of different algorithms (Hasan and Ng, 2010). For instance, the research of Yuan et al. (2018) and Florescu and Caragea (2017) showed that graph-based algorithms can outperform statistical algorithms. On the contrary, in the research of Sterckx et al. (2018) and Hasan and Ng (2010), statistical algorithms outperformed graph-based approaches.

Accordingly, the main objective of this thesis is to determine which unsupervised algorithm obtains the highest score on established evaluation metrics, for documents in the long-term healthcare domain. Furthermore, we hypothesize that combining different algorithms into an ensemble can outperform current state-of-the-art unsupervised algorithms.

We argue that there are three issues in the keyphrase extraction literature. Firstly, the evaluation methods in many studies are ill-defined because algorithms are evaluated on the keyphrases of a single annotater. As a result, algorithms are penalized for evaluation errors because keyphrases by a single person

[1] Automatic because no human annotators are necessary to extract keyphrases.

[2] In terms of terminology there is some ambiguity: some researchers refer to this field of research as keyword extraction, while most researchers refer to it as keyphrase extraction. Throughout this thesis, keyphrase extraction will be used because phrases of two or more words are common (Turney, 2000).

are usually not the only correct ones (Medelyan and Witten, 2006; Nguyen and Kan, 2007; Sterckx et al., 2018). The research by Sterckx et al. (2018) has shown promising results by utilizing multiple annotators, yet there is a lack of consensus regarding suitable inter-annotator agreement (IAA) measures. In fact, most researchers fall back on using pure categorization agreement measures. For example, Sterckx et al. (2018) and Jiang et al. (2009) utilized Fleiss' kappa, whereas Wan and Xiao (2008) utilized Cohen's kappa; both of which are suitable for categorization tasks. However, in this thesis, we propose the idea that using traditional categorization agreement measures is questionable because keyphrase extraction is *not* a pure categorization task.

Secondly, there is a discrepancy in previous studies if extracting keyphrases is harder for shorter or longer documents. Namely, Zhang et al. (2016) argues that short texts are the hardest to summarize because there is little data to calculate certain statistical features for each phrase. In contrast, Hasan and Ng (2014) argues that longer documents are harder because there are more potential keyphrases, which results in a larger search space.

Thirdly, a neglected area in the field is utilizing background knowledge to extract abstractive keyphrases. Essentially, abstractive keyphrases are keyphrases which are *absent* in the text, but effectively summarize the content of the article. We hypothesize that utilizing a domain-specific thesaurus may allow us to find suitable abstractive keyphrases.

Our methodology consisted of having three domain-experts as human annotators. As a result, we have a well-represented ground truth of keyphrases to evaluate the developed systems. Also, we introduce two new agreement measures that are suitable for keyphrase extraction. Furthermore, we implemented several existing unsupervised algorithms and combined these to create two ensemble methods. Also, we combined a domain-specific thesaurus with graph databases to extract suitable abstractive keyphrases.

The experimental results suggest that state-of-the-art algorithms provide rather unsatisfactory results for documents in long-term care. Nonetheless, our initial hypothesis indicates to be true since creating an ensemble algorithm can greatly improve the results across a wide variety of evaluation metrics. Specifically, our voting ensemble provided a relative improvement of 14.7% in terms of mean average precision (MAP)@5 with the best-performing state-of-the-art solution; these results were also statistically significant. Furthermore, the ensembles suggest being robust because they perform well on short and long documents. Additionally, we provide anecdotal evidence that graph databases can enrich the final set of keyphrases, by adding domain-specific information related to the document.

Structure of this thesis

This thesis is divided into seven chapters. Chapter 2 explains the business context and determines the objective of this research. Chapter 3 aims to give an overview of the current literature. Chapter 4 describes all the algorithms that are implemented in this research. Chapter 5 explains the experimental

setup, and Chapter 6 presents our experimental results. Finally, Chapter 7 concludes this research, discusses the limitations of this research, and gives concrete recommendations for future researchers in this field.

Chapter 2

Problem statement

In Section 2.1 the business context is given by describing the host company BearingPoint, and Vilans; a client for which this research is conducted. The potential value of this research is described in Section 2.2. Lastly, in Section 2.3 we state the main research objective.

2.1 Company background and context

Vilans is a knowledge institution in the Dutch long-term care. They have the responsibility of providing long-term healthcare providers (ranging from caregivers to managers) with the right information. Essentially, they serve as a bridge between the medical research community and healthcare professionals.

On the one hand, they validate scientific findings and translate this academic knowledge into understandable language for the healthcare professional. Subsequently, this knowledge is shared on their various knowledge websites. On the other hand, they collect information that healthcare professionals need to improve and accelerate decision making. As a consequence, they shape the research agenda of the academic world, so that this corresponds with the needs of the healthcare professionals. Figure A.1 illustrates this process.

Of course, after validating and translating the recent scientific developments this information has to be easily obtainable for the health care professional. To ensure this, Vilans established the following three strategic goals for 2019 until 2022:

- The right knowledge has to be at the right location; knowledge has to be tailored towards the professional.
- Digital transformation will be used as an aid for this.
- Strengthen the knowledge infrastructure (Web, 2019).

These are challenging endeavors because knowledge of Vilans is scattered among different knowledge platforms. To gain technical support they approached Bear-

ingPoint in 2019.[1] At the moment of writing this thesis (2020), BearingPoint is creating a knowledge graph, which links related documents to each other in a visual manner. The utilization of a knowledge graph can improve the current document management system, and could empower faster and more informed decision making for healthcare professionals.

2.2 Potential value of this research

In the previous section the benefits of implementing a knowledge graph for Vilans are explained, but how does one connect related documents in a knowledge graph? One possible method would be to extract the most relevant keyphrases for each document, which provides useful meta-data for the knowledge graph. In short, a keyphrase extractor could benefit Vilans in the following way:

- It could provide meaningful connections between documents with similar keyphrases.
- Keyphrases could serve as a useful index for searching the knowledge graph.
- It will provide the readers with a concise summary of the documents, so they can quickly decide if it is relevant for them.
- Useful for potential future NLP projects (e.g., features for a text classification algorithm).

One could argue keyphrases could be assigned manually by humans, but this would be a costly and time-consuming endeavor. In comparison, an automatic keyphrase extractor can quickly extract keyphrases for their current and future documents. Additionally, it provides a generic and explainable method of finding the most relevant keyphrases. In comparison, if one person assigns keyphrases these are considered to be highly subjective (Medelyan and Witten, 2006; Nguyen and Kan, 2007; Sterckx et al., 2018).

[1] BearingPoint is a consulting firm with a focus on management and technology consulting.

2.3 Research objectives

The goal of this research was to develop an algorithm that can extract keyphrases without any labeled data because this was unavailable at the start of this research. In short, we propose the following research goal:

Developing an unsupervised algorithm that can accurately extract keyphrases, for documents in the long-term care sector.

Chapter 3

Literature review

In the literature, there are a wide variety of techniques used for keyphrase extraction. In Section 3.1 we provide an overview of the most prevalent methods. Section 3.2 and 3.3 explain supervised and unsupervised techniques, respectively. Thereafter, in Section 3.4 we reflect on our observations from Section 3.2 and Section 3.3. More specifically, we provide limitations of the current state-of-the-art and report general discrepancies in the literature. Afterwards, in Section 3.5 the most popular evaluation approaches are discussed. In Section 3.6 common IAA in keyphrase extraction are discussed. Lastly, in Section 3.7 our literature review is concluded, and we propose six sub-questions that provide a framework for this research.

3.1 Available methods

The goal of this section is to illustrate the wide variety of techniques that are prevalent in the literature. Hasan and Ng (2014) provided a clear overview of all the possible methods. Of course, in recent years there have been advancements in this field, but the framework they suggested is still relevant. This literature review utilizes the framework they proposed, but some adjustments are made. Specifically, we ignore techniques that are obsolete and incorporate significant contributions in this field since the writing of their paper. As a result, we classify the keyphrase extraction literature in the following sections:

1. Supervised approaches:
   (a) task reformulation;
   (b) deep learning.
2. Unsupervised approaches:
   (a) statistical algorithms;
   (b) graph-based algorithms.

Difference between supervised and unsupervised methods

A fundamental distinction between supervised and unsupervised methods is that supervised methods require documents to contain keyphrases for every document, whereas the latter is more flexible; no existing keyphrases are necessary. Furthermore, supervised methods require to separate the data set in three parts: a training, validation, and test set. The training set is used to train the algorithms, the validation set is used to give an indication of the performance, and to optimize the hyperparameters. Finally, the test set is used to evaluate the algorithms. In contrast, unsupervised approaches do not require a training set. This property is especially interesting for this research considering the fact that documents of Vilans do not contain keyphrases for every article. Nevertheless, a literature review cannot ignore supervised methods because it is employed in many papers. In addition, it achieves state-of-the-art performance on many benchmarking data sets.[1]

Because supervised learning was not used for this research these techniques are not thoroughly explained in this literature review. The curious reader is suggested to read the original papers for a description of these algorithms. In comparison, the unsupervised methods implemented in this research are explained in Chapter 4.

3.2 Supervised approaches

3.2.1 Background

In general, given a set of N training examples of the form $\{(x_1, y_1), \ldots, (x_N, y_N)\}$ such that x_i is the feature vector (as described in Section 3.2.3) of the i-th example and y_i is its label (i.e, keyphrase or non-keyphrase), a supervised learning algorithm seeks a function $g : X \to Y$, where X is the input space and Y is the output space. In essence, the goal during the training phase it to minimize a loss function: $L(y_i, \hat{y_i})$, where y_i is the value according to the "ground truth" (hereafter ground truth) of the training example $(x_i,\ y_i)$, and $\hat{y_i}$ is the predicted value for the same training example.

Essentially, the idea of the loss function is to penalize bad predictions during the training phase. Therefore, an assumption is that the keyphrases used in the training phase can be assumed to be the ground truth. For example, consider the scenario where a model would predict a certain phrase as a keyphrase with a high probability. If this is actually a keyphrase the loss is small, but if this is not true the model made a significant error and should be penalized accordingly. In many NLP tasks including keyphrase extraction there is no ground truth because language is subject to interpretation. Thus, the ground truth is unattainable. As an alternative NLP applications rely on human annotations, which is often referred to as the "gold standard" (hereafter gold standard).

[1] For more information about these data sets: https://github.com/boudinfl/ake-datasets

3.2.2 Task reformulation

In task reformulation, keyphrase extraction can be formulated as a binary classification problem; the actual keyphrases can be encoded as positive examples, and non-keyphrases can be regarded as negative examples. One of the most significant contributions in this approach was made by Turney (2000), where he showed that the C4.5 decision trees can be applied for keyphrase extraction. A detailed explanation of the C4.5 decision tree can be found in (Larose and Larose, 2014, Chapter 8).

We want to emphasize that C4.5 decision trees are useful, but suffer from some shortcomings. Most notably, decision trees are prone to overfitting, which means that they tend to exactly or closely match a particular set of data points. As a result, decision trees usually fail to generalize, which means they perform poorly on the test set if this differs from the training set. This problem is especially problematic for keyphrase extraction because keyphrases from the training and test set could differ greatly if the annotated keyphrases are by different human annotators (Sterckx et al., 2018).

Decision trees failure to generalize has led to the rise of ensemble learning in a wide variety of applications including keyphrase extractions. Recently, there has been a growing interest in applying gradient boosted decision trees (Sterckx et al., 2018). An elaborate explanation of this method is beyond the scope of this literature review, but in essence, their variance is much lower than a classical decision tree, which allows for better generalization.

To illustrate this: Erdal and Karakurt (2013) compared both models on the same data set and concluded that gradient boosted trees can significantly outperform single decision trees. The reason for this is that it combines the prediction of multiple decision trees, so it is less susceptible to overfitting on the training data.

The main limitation of reframing keyphrase extraction as a binary classification task is that every keyphrase is considered to have the same relevance. However, certain keyphrases might also be more important than others, but binary classification disregards this important characteristic. This raises many questions as to whether reframing keyphrase extraction as a binary classification problem is a valid approach.

Learning to rank

An alternative solution that effectively solves the limitation in the previous paragraph, is to explore learning to rank techniques. These methods are often used for information retrieval purposes, where documents are sorted by their relevance for the query. For this reason, Jiang et al. (2009) experimented with learning to rank techniques, which resulted in Ranking SVM outperforming traditional binary classifiers.

3.2.3 Common features

The classifiers explained in Section 3.2.2 require useful features to identify each keyphrase as a keyphrase or a non-keyphrase. The features used in most studies fall into two categories. Firstly, there are within-collection features, which uses information from resources inside the original text. The most popular and best-performing single feature is term frequency-inverse document frequency (tf-idf) (Kim et al., 2013). Moreover, utilizing the position of a phrase has also led to good results (Hulth, 2003; Zhang et al., 2007). Furthermore, linguistic features (e.g., part-of-speech for each word) have also been exploited (Sterckx et al., 2018).

Secondly, there are external resource-based features, which uses information from resources outside the original text. For example, Yih et al. (2006) created a feature that checks if a candidate keyphrase is contained in a query log. The reasoning behind this is that keyphrases that appear often in user queries are more likely to be a keyphrase. Also, Berend and Farkas (2010) used WikiPedia, where a candidate keyphrase can have a binary value; 0 if it does not appear in the title or abstract of a WikiPedia page, or 1 if it does.

Furthermore, Medelyan and Witten (2006) implemented incorporating a domain-specific thesaurus by incorporating a node degree for each word. The node degree represents the number of thesaurus links that connect the term to other candidate phrases. The reasoning behind this feature is that, if a document describes a specific topic then it is likely related to the thesaurus term of this topic. For this reason, candidate keyphrases with a high node degree are usually likely to be a keyphrase.

External resource-based features may be useful for machine learning because it incorporates background knowledge from the domain. In general, a lack of background knowledge is one of the major limitations of many state-of-the-art algorithms (Hasan and Ng, 2014).

3.2.4 Deep learning

Deep learning methods are becoming increasingly important due to their demonstrated success at dealing with unstructured data. Recently, there have been many studies that focused on implementing various deep neural networks (DNNs) architectures for keyphrase extraction (Basaldella et al., 2018; Meng et al., 2017; Yuan et al., 2018; Zhang et al., 2016).

The approach by Zhang et al. (2016) consisted of creating a joint-layer recurrent neural network. In essence, this architecture consists of two hidden layers, and two output layers which were combined into one objective layer. In brief, the approach consisted of four steps. Firstly, the text is tokenized, so that each word is represented by one token. Secondly, the hidden layers are represented by word embedding representation, such that each input word is represented by continuous vector representation. An extensive explanation of word embedding is beyond the scope of this paper, but we refer to the work of Pennington et al. (2014). The first output layer of Zhang et al. (2016) his

recurrent neural network consisted of a boolean value; *true* if the token was part of a keyphrase, and *false* otherwise. The second output layer could contain five values: *single* (keyphrase consisting of one word), *begin* (beginning of a keyphrase), *middle* (neither beginning or end of a keyphrases), *end* (end of a keyphrase), and *not* (not a keyphrase).

The approach by Basaldella et al. (2018) also consisted of four steps. The first two steps are similar to the approach by Zhang et al. (2016): words are tokenized, and hidden layers are represented by word embeddings. However, it should be stressed that the actual architecture Basaldella et al. (2018) used is a Bidirectional Long Short-Term Memory Recurrent Neural Network (an alternative to the Recurrent Neural Network). The major advantage of this network structure is that it *can* also take the future context of a specific word into account, which is beneficial for many NLP tasks including keyphrase extraction (Basaldella et al., 2018). Lastly, the DNNs is connected to a fully connected hidden layer, which in turn is connected to a softmax output layer with three neurons for each word; no-keyphrase, beginning-keyphrase (the first token of a keyphrase), and end-keyphrase (the remaining tokens of a keyphrase). Therefore, the actual output layer used in both papers is different.

The main benefit in comparison to the techniques discussed in Section 3.2.2, is that DNNs can extract implicit information by using sophisticated model architectures. More specifically, this allows them to better understand the context of a piece of text; something which is known to be especially difficult for many machine learning algorithms.

Furthermore DNNs do not rely on extensive feature-engineering because they can utilize pre-trained word embeddings. This is a major advantage because feature-engineering is known to be time-consuming and domain-specific. For instance, Turney (2000) experimented with 110 different features, but only nine features were considered to aid the models performance.

The results of Zhang et al. (2016) and Basaldella et al. (2018) appear promising, but their research reveals two major shortcomings. Firstly, both papers evaluated their results using only one data set, whereas most researchers in this field used several data sets to evaluate their algorithms. Secondly, their algorithms are only evaluated on relatively short documents. Basaldella et al. (2018) used the Inspec data set, which consists of abstracts ranging from 23 to 338 tokens on the test set (Hulth, 2003), whereas Zhang et al. (2016) evaluated the model on Twitter data where each tweet has a character limit of 140. Thus, it becomes unclear from their research how deep learning performs on longer documents.

Extracting absent keyphrases

Although Zhang et al. (2016) and Basaldella et al. (2018) used the power of DNNs the approach taken by Meng et al. (2017) is radically different. Unlike the previous supervised learning methods discussed, their approach was able to accurately extract *abstractive keyphrases* by using a recurrent neural networks (RNN).

Abstractive keyphrases are keyphrases that are not present in the document, but still, accurately summarize the content of an article. It should be recognized that this is useful because previous methods were unable to deal with these keyphrases, but they do constitute 32.25% to 57.99% of the keyphrases of public data sets (Meng et al., 2017). The reason for this is that human-annotators tend to first read a document, after which they scan the document again to see if there are any relevant keyphrases. However, if there are no relevant keyphrases in the document, humans rely on their own domain knowledge and creativity, to create a relevant keyphrase that is not present in the document.

The architecture relied on a recurrent neural network, but used a encoder-decoder model, so that content of a source document could be represented through a hidden representation with an encoder, and corresponding keyphrases could be extracted with the decoder. In general, this architecture provides better performance than both architectures used by Zhang et al. (2016) and Basaldella et al. (2018). Specifically, the model proposed by Meng et al. (2017) can extract abstractive keyphrases by not limiting the size of the vocabulary to a fixed threshold. Therefore, it incorporates a certain probability of copying a keyphrase from the source text of the *full corpus*. Thus, ensuring the model can also extract abstractive keyphrases.

The results appear promising because the model can extract up to 20% of absent keyphrases from the gold standard. For this reason, their architecture outperforms traditional methods across six benchmarking data sets of various lengths. In contrast, most other methods in the literature are not so consistent across a wide variety of data sets.

Extracting an optimal number of keyphrases

A disadvantage of all previously discussed models is that they are unable to extract an "optimal" number of keyphrases. However, during evaluation, the actual number of keyphrases according to the gold standard may depend on many factors. For instance, the length of the article, the domain, and the person who annotates the keyphrases can all influence this (Yuan et al., 2018). A discussion of why this is problematic is discussed in Section 7.2.

Recently there have been attempts to determining an optimal number of keyphrases for a document. For example, Yuan et al. (2018) proposed a self-terminating decoding strategy by dynamically estimating the proper size of the target phrase set.

Limitations of deep learning

We want to emphasize that DNNs can achieve state-of-the-art results, but have several limitations. The major drawback with regards to this research is that it requires much-annotated training data. This may be illustrated by the fact that Meng et al. (2017) used 527 830 documents to train their DNNs.

Ye and Wang (2018) proposed a novel solution to this problem by introducing semi-supervised learning to this field. Specifically, they performed the following: first, they tagged unlabeled data with synthetic keyphrases obtained from unsupervised keyphrase extraction methods. Afterward, they combined labeled samples in the target domain for training. Their experimental results proved promising because they outperformed several unsupervised approaches, but required much less labeled data than the DNNs previously discussed. However, it is worth mentioning that they still used a considerable amount of labeled data; in fact, they used 40 000 documents with annotated keyphrases, and 400 000 documents were synthetically created by using unsupervised methods.

3.3 Unsupervised approaches

In general, the performance of many supervised (and semi-supervised) approaches is considerably better. Nevertheless, unsupervised methods provide two major advantages: they do not require any labeled data, and are scalable among a wide variety of languages and domains. In unsupervised learning, keyphrase extraction is reframed as a ranking problem, where the weight of a word determines its importance. In short, there are two popular approaches in the literature: statistical and graph-based algorithms.

3.3.1 Statistical approaches

In statistical algorithms, the weight of a word is based on the number of occurrences and in some techniques on its relative importance. It should be recognized that they can be very effective; for example, tf-idf is still widely used as an effective baseline (Florescu and Caragea, 2017; Hasan and Ng, 2014; Yuan et al., 2018). In fact, in many papers tf-idf turned out to the best-performing unsupervised approach (Hasan and Ng, 2010; Sterckx et al., 2018). A description of the statistical algorithms relevant for this research are given in Section 4.1.

3.3.2 Graph-based ranking

Graph-theory has a wide variety of applications including keyphrase extraction. One seminal paper by Mihalcea and Tarau (2004) proposed an innovative algorithm called TextRank, which is an algorithm inspired by Google's Pagerank algorithm. Although the applications of TextRank and PageRank are different, the basic idea is the same; they are both iterative based algorithms that propose an optimal ranking in the end. PageRank offers an optimal ranking consisting of Web pages with their ranks in descending order, but TextRank offers an optimal ranking consisting of keyphrases, which are ranked in descending order. A description of TextRank is given in Section 4.2.3.

After this groundbreaking paper, an increasing number of studies have found that graph-based algorithms can outperform classical statistical-based algorithms (Bougouin et al., 2013; Sterckx et al., 2015; Wan and Xiao, 2008).

The main limitations of this algorithm are that: (1) words that are strongly connected are usually highly ranked; however, this does not mean they are related to the major topics in the article, and (2) good keywords should cover all the topics of the article; however, graph-based approaches could focus on only one major topic (Liu et al., 2010, p366-377). Nonetheless, one could argue that the second limitation also applies to statistical-based approaches.

Topic-based clustering

The seminal contribution of TextRank led many researchers to combine clustering techniques with graph-based algorithms to ensure all main topic(s) are included as a keyphrase. There exists a considerable body of literature on topic-based clustering starting around 2008. One of the most recent significant findings by Boudin (2018) focused on extracting keyphrases using multipartite graphs In brief, this approach addresses the disadvantages of TextRank by ensuring topical diversity in the final set of keyphrases. A description of multipartite graphs is given in Section 4.2.5.

Position-based ranking

Florescu and Caragea (2017), an authority on keyphrase extraction, mentioned the fact that one of the advantages of TextRank is also a limitation. Namely, it does not exploit the word-frequency and position for each word, while these have been useful features for many supervised learning tasks (Hulth, 2003; Zhang et al., 2007). Thus, the paper by Florescu and Caragea (2017) focused on forming a "biased" TextRank where words that are more frequent, and occur in the beginning of a document have a higher initial weight. For this reason, these words are more likely to be extracted as keyphrases.

Ensembles

Surprisingly, there has been much research on individual unsupervised algorithms, but to our knowledge there has been little research on combining these models into an unsupervised ensemble. Nonetheless, ensemble methods can also be used through manipulating the output targets (Dietterich, 2000). Statistical-based and graph-based algorithms have a completely different approach to extracting keyphrases, and in general, ensemble methods can provide better results when there is a significant diversity among the algorithms (Dietterich, 2000). Also, previous research has shown that an ensemble is often more accurate than any of the single classifiers in the ensemble (Opitz and Maclin, 1999). For this reason we hypothesize that combining these algorithms into one ensemble can lead to a more diverse set of keyphrases, which could improve the results.

We strongly believe this because the aim of keyphrase extraction is to find the most relevant keyphrases k; however, it can be hard to find k using a single model because the search space is large Hasan and Ng (2014). As a result, many researchers have become increasingly interested in utilizing linguistic features to

limit the number of keyphrases candidates (Hulth, 2003; Mihalcea and Tarau, 2004). Although this effectively removes many unnecessary phrases, the experimental results of Sterckx et al. (2015) illustrated that these techniques alone are usually not sufficient; many k are still considered irrelevant in comparison to the gold standard.

Although ensemble learning is often applied in supervised settings it can still be applied in an unsupervised setting. To our knowledge, only Ye and Wang (2018) combined two unsupervised algorithms. Namely, in order to create synthetic keyphrases they used both tf-idf and TextRank to extract the top five keyphrases for a document. Afterwards, the union of the keyphrases was taken as the final set of keyphrases for this document. However, we argue that considering the union is quite a strict requirement because this often leads to much less than five keyphrases.

Unsupervised ensemble methods have also been utilized in another NLP task, for instance, Wan (2008) used it for Chinese sentiment analysis. Chinese sentiment analysis is a challenging task because there is a lack of labelled data in this language. Therefore, similar to this research applying supervised learning is infeasible. In short, their approach focused on a two-step approach:

1. Translating Chinese reviews into English by using publicly available machine translation resources. Thereafter, the sentiment polarity of English reviews were leveraged by using English resources.

2. Utilizing available resources for Chinese sentiment polarity.

Afterwards, both approaches were blended to form an ensemble. An extensive discussion of the approaches they used is beyond the scope of this thesis, but one approach consisted of majority voting. Meaning that the polarity tag, which received the most votes is chosen as the final polarity tag of the review. Majority voting is a popular approach to blend several models, which is also used for many supervised learning algorithms (e.g., classification in Random Forest).

We strongly believe that majority voting, and different approaches could also be applied for keyphrase extraction. Especially, because one of the major pitfalls of keyphrase extraction is finding creative solutions to effectively deal with the large number of candidate keyphrases. We hypothesize that casting a majority voting could offer more "confidence" regarding the final k, because a keyphrase only ends up in the k of the ensemble model, if it is considered relevant by multiple individual algorithms.

3.4 Discussion of the current state-of-the art

In general, supervised learning usually outperform unsupervised methods. For instance, a supervised learning algorithm obtained the best performance in the

SemEval2010 competition[2]. More recently, state-of-the art supervised techniques like gradient-boosted-trees and DNNs outperformed several unsupervised methods on multiple data sets (Meng et al., 2017; Sterckx et al., 2018; Yuan et al., 2018).

Limitations of supervised learning

As long as manually assigned keyphrases are available supervised approaches can be successful; however, in many real-life applications including this research, there are many constraints. In brief, there two major limitations of the supervised techniques discussed in Section 3.2.2, which are: 1) they rely on feature-engineering, but useful features are usually domain or text specific; 2) it requires labeled data, which in the real-world can be cumbersome and costly to acquire. DNNs effectively solves the first drawback, they usually require more data, so the second drawback is still relevant.

Both problems pose some serious problems with regards to this research. Firstly, even though the documents of Vilans focus on long-term healthcare, they still focus on a wide variety of topics in this domain, as is discussed in Appendix C. As a result, useful features for one specific sub-domain could prove to be useless for a different sub-domain. Semi-supervised methods could potentially partly solve this problem, but they still require some labeled data in the target domain.

To create the gold standard we rely on high-quality annotations by domain-experts. Instead, many studies in this field utilize author-generated keyphrases (Florescu and Caragea, 2017; Meng et al., 2017; Mihalcea and Tarau, 2004; Yuan et al., 2018). One of the major drawbacks to adopting this approach is that keyphrases assigned by one annotator are usually not the only correct ones (Medelyan and Witten, 2006; Nguyen and Kan, 2007; Sterckx et al., 2018). To illustrate this, we refer to the study of Sterckx et al. (2018) who used multiple annotators for each article. From their results we can state that if they were to restrict keyphrases, to those selected by 50% of the annotators, they would retain less than 5% of all keyphrases.

Consequently, imagine we would train a machine learning model on keyphrases by a single annotator, as is done in many studies. As a result, the performance on the evaluation and test set is heavily reliant on keyphrases from the annotator. Namely, if these keyphrases are annotated by a different annotator the performance could be disappointing. This raises many questions about whether supervised learning should be employed for keyphrase extraction.

Limitations of both supervised and unsupervised approaches

One of the major limitations of most current state-of-the-art algorithms is that most approaches are unable to extract an "optimal" number of keyphrases.

[2]SemEval is a yearly NLP competition to investigate the current state-of-the-art performance on a wide variety of problems. The 2010 competition focused on keyphrase extraction, among others.

However, during evaluation, the actual number of keyphrases according to the gold standard can depend on many factors, as was discussed in Section 3.2.4.

Another major shortcoming of current algorithms is that they are unable to deal with abstractive keyphrases. To our knowledge, only Meng et al. (2017) and Yuan et al. (2018) provided a solution for this limitation. However, a shortcoming of both approaches is that it requires a large training set, so they are *not* applicable to most businesses that have none or little labeled data.

Discrepancies in the literature

From studying the state-of-the-art algorithms we can conclude there are two main discrepancies regarding the performance of the algorithms. Firstly, it remains unclear if extracting keyphrases is harder for shorter or longer documents. Namely, Zhang et al. (2016) argues that short texts are the hardest to summarize because there is little data to calculate certain statistical features for each word. For example, in short texts, words are more likely to have the same word frequency, and also word co-occurrences (discussed in Section 4.2.3) provide little value. In contrast, Hasan and Ng (2014) argue that longer documents are harder because there are more potential keyphrases, which results in a larger search space.

Secondly, there is a lack of consensus regarding the effectiveness of different unsupervised algorithms (Hasan and Ng, 2010). For instance, the research of Yuan et al. (2018) and Florescu and Caragea (2017), showed that graph-based algorithms can outperform tf-idf. On the contrary, in the research of Sterckx et al. (2018) and Hasan and Ng (2010), tf-idf actually outperformed graph-based algorithms.

3.4.1 Shortcomings in the literature

In short, according to us, there are three major shortcomings in the literature. Firstly, as discussed in 3.3, to our knowledge there has been no research on creating an ensemble unsupervised algorithm.

Secondly, another shortcoming is that the best of our knowledge, all unsupervised algorithms are unable to extract absent keyphrases. This is not surprising because extracting abstractive keyphrases is a challenging task, and usually domain-specific. However, we strongly believe that methods to solve this problem are required to improve the current state-of-the-art.

3.5 Tradeoffs in evaluation

In general, there are two options in the evaluation phase for keyphrase extraction. These are:

- obtaining a gold standard, and using automated evaluation metrics;
- using human evaluation.

The steps with regards to obtaining the gold standard were explained in Section 3.4. An alternative solution is to conduct human evaluation, such that humans denote if each keyphrase can be considered relevant. This method is often used in information retrieval; a field which shares many of the same characteristics of keyphrase extraction.

It should be stressed that there is a discrepancy in results in the existing literature. For example, Turney (2000) used both evaluation methods and found out that their best solution was unable to achieve above 30% precision on the human evaluation set, yet by conducting human evaluation in a later stage they determined that more than 60% of the keyphrases can be considered relevant. This significant discrepancy is illustrated in Table 3.1, where the benefit and drawback(s) of each method are discussed. The drawbacks of using a gold standard will be discussed in Section 7.2.

Table 3.1: The benefits and drawback(s) of the most common evaluation metrics.

method	benefit	drawback(s)
gold standard	easy to compare different algorithms.	(1) evaluating abstractive keyphrases; (2) the exact match restriction; (3) the number of candidate keyphrases (especially for long pieces of text).
human evaluation	better at capturing the quality of the keyphrases.	costly and time-consuming to compare different algorithms.

From the literature, we can conclude most studies decide to utilize a gold standard (Florescu and Caragea, 2017; Mihalcea and Tarau, 2004; Sterckx et al., 2018; Yuan et al., 2018). Although this method has limitations, it does allow us to effectively compare different methods. In contrast, comparing several methods with human evaluation would be time-consuming for the annotators because each individual algorithm has a unique set of keyphrases that requires evaluation.

3.6 The importance of agreement measures

Section 3.4 introduced the notion that keyphrase extraction is a subjective task. However, this concern does not only apply to keyphrase extraction because many other NLP problems suffer from the same problem. Therefore, to measure to which extent the annotations are similar IAA is often used (Eisenstein, 2018). A high agreement indicates there is much overlap between the annotations (i.e., the

annotators agree often). Conversely, a low inter-annotator agreement indicates there is not much overlap between the annotations (i.e., the annotators often disagree).

The gold standard cannot be considered the real ground truth, but it should at least resemble it. Therefore, if multiple annotators are used it is important that annotators commonly agree because this shows two qualities of the gold standard:

1. The higher the agreement measure, the more plausible it is to assume the annotations can be considered as ground truth.

2. The higher the agreement measure, the more likely it is that the results of the annotation tasks are reproducible; i.e., other annotators would annotate similar keyphrases. In contrast, a low inter-annotator agreement measure might indicate an ill-defined task description, or that the annotation task should have stricter guidelines.

3.6.1 Inter-annotator agreement in keyphrase extraction

There is a tendency in the keyphrase extraction literature to focus on algorithmic development rather than the granularity of keyphrases used for training supervised models and evaluation. In fact, many studies in this field do *not* report agreement measures.

We propose two reasons for this. Firstly, many researchers rely on author-generated keyphrases of scientific papers, so calculating inter-annotator agreement is impossible. Secondly, calculating the inter-annotator agreement in keyphrase extraction is less straightforward than many other NLP tasks, which focus on categorization. Instead, in keyphrase extraction calculating the inter-annotator agreement with multiple annotators is non-trivial due to the possible number of keyphrases, the possible word combinations, and the variable number of keyphrases.

Nonetheless, the lack of quantitative measuring inter-annotator agreement for keyphrase extraction is a significant limitation in this field. However, there have been some attempts to measure IAA between multiple annotators; most notably by Augenstein et al. (2017); Jiang et al. (2009); Sterckx et al. (2018). These researchers reported the Fleiss' kappa, which is often used to calculate the agreement between three or more annotators in categorization.

Categorization is a task where annotators can choose between a fixed number of classes. However, a weakness with this approach is that keyphrase extraction is inherently not a pure categorization task. In contrast, we argue that keyphrase extraction differs from categorization, and is similar to a task where categorization meets unitizing.

Unitizing tasks are tasks where annotators have to define which elements are relevant and the corresponding position. Mathet et al. (2015) mentioned that due to the lack of agreement measures for unitizing many researchers fall

back on measures for categorization. They highlight four reasons why this approach is questionable, three of which we believe are also relevant for keyphrase extraction:

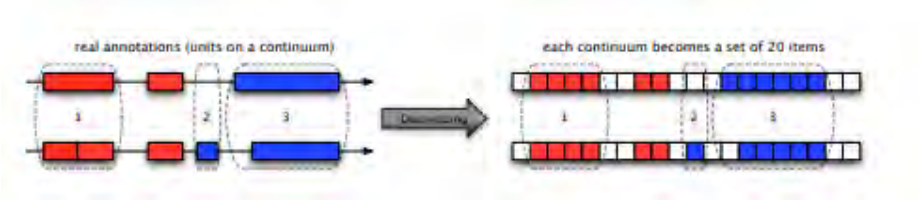


Figure 3.1: Examples of possible disorders. Source: (Mathet et al., 2015, p. 445)

1. Annotators agreeing on a keyphrase are given as much weight as a disagreement, and longer documents have more agreement due to there being more agreement about non-keyphrases. The reason for this is that reframing keyphrase extraction to a categorization task yields to discretization. Thus, non-relevant keyphrases are assigned a value of zero, and relevant keyphrases are assigned a value of one. Consequently, phrases that are considered non-keyphrases are assigned the value zero (not-relevant), therefore, annotators agree on these terms. Especially in longer documents where there are many non-keyphrases, this can lead to a misrepresentation of the actual agreement.

2. Two separate keyphrases that are adjacent are considered the same as one keyphrase constituting of the whole part. This is clearly illustrated in zone 1 of Figure 3.1, where the annotater at the top assigned one keyphrase, and the annotator at the bottom annotated these as two separate keyphrases. This is problematic because discretization leads to these being the same. Although they are similar they are not the exact same.

3. Positional disagreement and negative disagreement are given the same weight. For example, consider zone 2 of Figure 3.1 where there is a disagreement about a phrase being a keyphrase. This can be considered a severe disagreement, but this is given the same weight as a slight positional disagreement as can be seen in zone 3 of Figure 3.1.

From these arguments, we can conclude that an agreement measure that focuses on positional instead of categorical disagreement is more suitable for this task. Although these IAA metrics have not been studied in keyphrase extraction, there has been recent research on finding appropriate IAA measures for similar NLP tasks such as named entity recognition (Mathet et al., 2015). Nevertheless, to our knowledge, this method has not been implemented to keyphrase extraction.

3.7 Concluding remarks

This research primarily focuses on accomplishing the research goal. However, from the literature review, we can determine there are several pitfalls in previous studies that could influence the accomplishment of this research goal.

3.7.1 Evaluating keyphrase extraction

In general, determining the gold standard is non-trivial for many subjective NLP tasks. From the literature review, we can conclude this is especially the case for keyphrase extraction, due to the possible number of keyphrases, the possible word combinations, and the variable number of keyphrases. Furthermore, keyphrases assigned by one annotator are usually not the only correct ones (Medelyan and Witten, 2006; Nguyen and Kan, 2007; Sterckx et al., 2018).

However, as discussed in Section 3.6.1, even if multiple annotators are used a challenging area in this field is determining suitable agreement measures. In short, we can conclude that determining the granularity of annotated keyphrases with current state-of-the-art IAA measures is inadequate. Thus, we propose our first sub-question (SQ):

SQ1: What can be considered the gold standard of keyphrases for each document, and how can we determine the granularity of these keyphrases?

After establishing the gold standard it is necessary to evaluate the algorithms in a quantitative manner. However, as discussed in Section 3.5 many researchers take a different approach in the evaluation phase. The main reason for this is that suitable evaluation metrics depend on the experimental setup. Therefore, we propose the second sub-question of this research:

SQ2: Which quantitative evaluation metrics are appropriate for the developed algorithms?

3.7.2 Performance of the algorithms

The main research objective is to develop an algorithm that can accurately extract keyphrases in the long-term care; however, according to Hasan and Ng (2010) the performance of many algorithms relies on certain statistical characteristics of a data set. Therefore, we require to know to which extent state-of-the-art unsupervised algorithms obtain satisfactory results for our data set, which leads us to following third sub-question:

SQ3: How accurately can unsupervised algorithms extract keyphrases in long-term care, measured against the gold standard?

Furthermore, as discussed in Section 3.4 there is a lack of consensus regarding the effectiveness of different unsupervised algorithms. Additionally, for rea-

sons discussed in Section 3.3.2 we hypothesize that an unsupervised ensemble can outperform each individual algorithm. Thus, we propose our fourth sub-question:

SQ4: Which unsupervised algorithms achieves the best results on the established evaluation metrics? And can an unsupervised ensemble method outperform each individual algorithm?

Also, as mentioned in Section 3.4 there is a discrepancy in previous studies if extracting keyphrases is harder for shorter or longer documents. Appendix C illustrates that Vilans has a combination of short and long documents, so it is essential that the best-performing algorithm is robust to changes in the search space. Thus, we propose our fifth sub-question:

SQ5: Does the length of a document have an effect on the performance of the unsupervised algorithms?

As is discussed in Section 3.4 extracting abstractive keyphrases is a challenging endeavor in this field. The research of Yuan et al. (2018) and Meng et al. (2017) focused on extracting abstractive keyphrases, but this solution is only applicable if a large training set is available. For this reason, we argue that this solution is not scalable to most practical applications. Nonetheless, we strongly believe abstractive keyphrases are required for effective summarization. Therefore, this research investigates to what extent it is possible to extract abstractive keyphrases with unsupervised algorithms. We argue that this approach is more scalable than the approaches by Yuan et al. (2018) and Meng et al. (2017) because it does not rely on a large annotated corpus, and within many domains, a domain-specific thesaurus is often available. Thus, this leads us to our last sub-question:

SQ6: Can the use of a domain-specific thesaurus supplement the best-performing unsupervised algorithm, by extracting domain-specific abstractive keyphrases?

These sub-questions are used as a framework for this research. After answering these sub-questions we can determine if the research goal discussed in Section 2.3 is accomplished.

Chapter 4

Algorithms description

The goal of this section is to explain the algorithms that are implemented for this research. In Section 4.1 the statistical-based models are explained. In total, there are two statistical algorithms we will use for this research: term frequency (tf) and tf-idf, which are described in Section 4.1.1 and 4.1.2, respectively. Thereafter, the graph-based algorithms are explained, which are TextRank, PositionRank, and MultipartiteRank. Finally, two new ensemble algorithms we propose are explained.

The input for the term-frequency and graph-based approaches are the same; they both require a document as input and their output is the top n keyphrases in descending order based on the frequency or weight of each keyphrase. Similar to the other approaches tf-idf requires a document as input, but also requires a weight for each term. Section 4.1.2 explains how these weights are determined.

4.1 Statistical algorithms

To evaluate sophisticated algorithms we also require baseline algorithms. These simpler algorithms can be used as a reference point for evaluating more sophisticated algorithms.

4.1.1 Term frequency

To obtain a baseline algorithm, we chose to use term-frequency, which simply selects keyphrases based on their frequency. Each term constitutes of a phrase, which can be unigram, bigram or trigram. A difference between using unigrams and bigrams/trigrams is that in the latter case word combinations are taken into consideration.

It is worth mentioning for this algorithm stopwords are deleted because these are too generic and are not suitable for summarizing an document. Furthermore, words are lowercases, so they have the same representation. Next, we

tokenize the words, such that if we would only consider bigrams and we have the sentence: "preventie van dementie blijft een relevant onderwerp", we would obtain the bigrams given in table 4.1. All the bigrams in the omitted columns, are omitted because stopwords are ignored. It is a rather naïve model, but could still be useful because frequency tends to be important in how humans summarize information (Nenkova and Vanderwende, 2005).

Table 4.1: The bigrams and omitted bigram for the example sentence given in the above paragraph.

bigrams	omitted bigrams
"preventie dementie"	"preventie van"
"dementie blijft"	"van dementie"
"blijft relevant"	"blijft een"
"relevant onderwerp"	"een relevant"

Term frequency is represented by a matrix with dimensions $N \times T$, where the N correspond to the number of document in the corpus, and T are the number of *unique* terms, where stopwords are usually ignored. Essentially, each document is a count vector containing only natural numbers. An example of a term frequency representation is given below, where $tf_{t,d}$ is the frequency of term t in document d.

$$\mathbf{tf} = \begin{bmatrix} tf_{1,1} & tf_{1,2} & \cdots & tf_{1,T} \\ tf_{2,1} & tf_{2,2} & \cdots & tf_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ tf_{N,1} & tf_{N,2} & \cdots & tf_{N,T} \end{bmatrix}$$

After the term-frequency is determined for each document in the corpus, the next step is to extract the most frequent keyphrases. This done by extracting the top k terms with the highest frequency for each document, where k is the number of keyphrases we want to extract for the document.

4.1.2 Term frequency-inverse document frequency

The previous section dealt with frequency-based approaches, but an assumption underlying these methods is that a term that appears ten times more often is also ten times more important; however, in most scenarios this assumption is invalid. Alternatively, inverse document frequency can be incorporated to assign a weight to each term to identify its importance. This method is called tf-idf because it combines two important properties that contribute to the importance of a term: its frequency and informativeness.

In Equation 4.1 the formula for tf-idf is given, recall that M is the total number of documents in the corpus. Furthermore, df_t is the document frequency

of the term t, which is defined as the number of documents that contain the term t.

$$idf_t = \log_{10}\left(\frac{N}{df_t}\right) \tag{4.1}$$

It should be mentioned that each document in the corpus N has the same value for the term t. For example, $idf_{1,T} = idf_{2,T} = ... = idf_{N,T}$. Consequently, we can obtain the obtain the tf-idf values for every document in the corpus. This is done by multiplying the frequency of term t in document d by the inverse document frequency of the term t. An example for a matrix with dimensions $N \times T$ is provided below, note that $\circ$ is the hadamard product and *not* the matrix product.

$$\textbf{tf-idf} = \begin{bmatrix} tf_{1,1} & tf_{1,2} & \cdots & tf_{1,T} \\ tf_{2,1} & tf_{2,2} & \cdots & tf_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ tf_{N,1} & tf_{N,2} & \cdots & tf_{N,T} \end{bmatrix} \circ \begin{bmatrix} idf_{1,1} & idf_{1,2} & \cdots & idf_{1,T} \\ idf_{2,1} & idf_{2,2} & \cdots & idf_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ idf_{N,1} & idf_{N,2} & \cdots & idf_{N,T} \end{bmatrix}$$
$$= \begin{bmatrix} tf\text{-}idf_{1,1} & tf\text{-}idf_{1,2} & \cdots & tf\text{-}idf_{1,T} \\ tf\text{-}idf_{2,1} & tf\text{-}idf_{2,2} & \cdots & tf\text{-}idf_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ tf\text{-}idf_{N,1} & tf\text{-}idf_{N,2} & \cdots & tf\text{-}idf_{N,T} \end{bmatrix}$$

After calculating the hadamard product each document is represented by a vector of tf-idf weights, where each vector consists only of positive real numbers. The final step is to extract the most frequent keyphrases, which is done by extracting the top k terms with the highest tf-idf weight, where k is the number of keyphrases we want to extract for the document.

Although frequency-based features are useful they do suffer from constraints. Most notably, they do not consider synonyms, which can be quite problematic because most authors often use synonyms to avoid repetition. Another point that can be made is that they are context-independent, which means that these approaches are unable to capture the semantics of a language.

Lastly, it is worth mentioning that the inverse document frequency should be calculated on a large corpus, and on the same domain. Otherwise, the weights would be misrepresented.

4.2 Graph-based algorithms

4.2.1 Notation

Table 4.2: Formal notation used in this section

graph-based approaches	
G	mathematical graph
V	vertices of the mathematical graph
E	undirected edges of the mathematical graph
v_i	ith vertex
e_{ij}	an undirected edge connecting vertex i to vertex j
$\|Adj(v_i)\|$	the vertices that are adjacent to v_i
$S(v_i; t)$	weight of vertex v_i at time step t
d	damping factor

4.2.2 What is a mathematical graph?

Formally, a graph is specified by a set of vertices V and by edges E, such that we have a graph: $G = (V, E)$. These edges can be *directed* or *undirected*; a directed edge means there is an asymmetrical relationship between two vertices, and an undirected edge means there is a symmetric relationship. An example of both types of graph can be seen in figure 4.1.

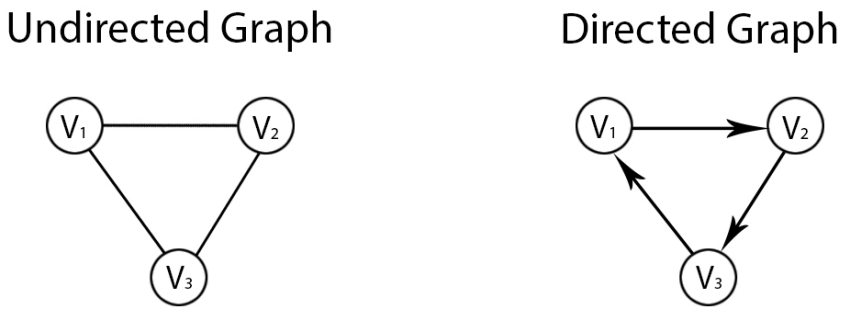


Figure 4.1: Examples of directed and undirected graphs. Adapted from "Example Local Clustering Coefficient on an Undirected Graph,". Retrieved June 30, 2020, from https://www.e-education.psu.edu/geog597i_02/node/832.

For this research graphs are beneficial because they allow us to draw relationships between words that could be related. Subsequently, we can extract phrases that are relevant, but might have a low term-frequency and/or inverse document frequency.

4.2.3 TextRank

Intuition of TextRank

The intuition behind TextRank comes from the PageRank algorithm; webpages are considered important if they have many links from other websites. It should be mentioned that not only the number of incoming links are important, but also the quality of that link. For example, if one webpage has little incoming links one might assume it is not important, but if one of the incoming links is very important, the webpage could actually end up becoming important. The importance of a webpage is determined by its weight; the higher the weight the more important the webpage is.

It should be mentioned that on the Web it is rather straightforward to determine the relationship between webpages. For example, consider the case where webpage A mentions webpage B, but the opposite is not true; that is to say webpage B does not mention webpage A. This would mean that the vertex representing webpage A would have a directed edge to webpage B, but the opposite is not true; that is to say, there is no directed edge from webpage B to webpage A.

Drawing directed relationships between words is less straightforward because words in a document cannot "recommend" each other. Nonetheless, Mihalcea and Tarau (2004) experimented with directed graphs by drawing a directed edge between two vertexes if one token appears after another. The intuition would be that one word "recommends" the following word; however, the results proved disappointing even by reversing the direction of the edges.

Even though words cannot necessarily "recommend" each other they can still be related by "co-occurring" between a specific window. The intuition behind this is that related words tend to be close to each other in the text, and infrequent words could have an incoming link by their more frequent neighbor. In this scenario, an infrequent word has an incoming link from a frequent word, if they co-occur within a predetermined window. Thus, because the infrequent words have an important neighbor, they could end up getting a high weight and be selected. As previously mentioned, in the TextRank algorithm the edges are undirected edges, which means that the PageRank algorithms would also need to work on undirected graphs, but experiments show this to be true (Gleich, 2015; Mihalcea and Tarau, 2004).

Description of TextRank

The TextRank algorithms requires a document as input. Thereafter, a word-graph is constructed, by initializing every *unique* word in the document as a

vertex if it passes one basic requirement: it passes the *syntactic filter*. The syntactic filter checks the *part of speech* of each word and acts as a tool to filter irrelevant words. According to the experiments of Mihalcea and Tarau (2004) and Hulth (2003), adding linguistic knowledge significantly improves the accuracy of the model by ignoring non-relevant words. Usually, only noun phrases are considered because humans tend to use noun phrases to summarize texts.

Note that $S(v_i; t)$ is the PageRank value of vertex i at time step t. Initially, for each vertex: $S(v_i; 0) = 1$, for $v_i \in V$. Thereafter, undirected edges between the words are created through a so called *co-occurance window*. More specifically, two vertixes are connected if their corresponding *lexical units* co-occur within a window of maximum N lexical units. Lexical units are defined to be words (including stopwords) and punctuation marks. Mathematically, vertex v_i is connected to vertex v_j through an undirected and unweighted edge e_{ij}, where $e_{ij} \in E$.

After the graph is initialized, the connections between words are made, and each vertex has an initial weight, the next phase of the algorithm starts. The goal of this stage is to find the most important words, which is calculated by adjusting the weight is an iterative manner. More specifically, at each time step, the weight of every vertex is adjusted using Equation 4.2. Recall that $|Adj(v_i)|$ are the number of vertices that are adjacent to vertex i. Note that two vertices are considered to be adjacent, if and only if there is an undirected edge connecting the two vertices. Also, note that $S(v_i; t+1)$ is the weight of vertex V_i at the next time step (t+1), and $S(v_j; t)$ is the weight of vertex V_j at the current time step.

Furthermore, let d be a damping factor which is usually set to 0.85 (Florescu and Caragea, 2017; Mihalcea and Tarau, 2004). The damping factor ensures we do not get stuck in cycles of the graph, by including in our model a certain probability of jumping from a given vertex to another random vertex. This is relevant for language modeling because word references can appear randomly (Mihalcea and Tarau, 2004).

$$S(v_i; t+1) = (1-d) + d \cdot \sum_{j \in Adj(v_i)} \frac{1}{|Adj\ (v_i)|} S(v_j; t) \tag{4.2}$$

According to the experiments of Mihalcea and Tarau (2004) the TextRank algorithm usually converges after 20-30 iterations , but to ensure proper convergence we can also define an error rate called epsilon. This means we assume convergence when $|S(v_i; t+1) - S(v_i; t)| < \epsilon$ for each $v_i \in V$, where ϵ is commonly set at 0.0001 (Mihalcea and Tarau, 2004) or 0.001 Florescu and Caragea (2017).

Next, words are sorted in descending order based on their PageRank weight. Afterward, words can be combined if and only if they are both in the top k keyphrases and are adjacent in the document. Their weights are normalized, and if they are still in the top k after normalizing, they are selected. With regards to selecting the top keyphrases there are two options; the first

one is simply to select the top k keyphrases, but the alternative is more flexible by selecting one-third of the number of vertexes. The benefit of the second approach is that it is more representative of the actual number of keyphrases in the text because it relies on the number of potential keyphrases. However, it could also lead to an unnecessary number of keyphrases for each document, so depending on the application one of these two options can be chosen.

Thus, to summarize: the algorithms start by constructing a word-graph, where each vertex consists of a word that passed the syntactic filter. Initially, each vertex has a weight of 1, but then we utilize equation 4.2 for each vertex until the PageRank algorithm converges. Next, we sort the words in descending order based on their PageRank weight. Afterward, we combine adjacent words and normalize their weights. Lastly, we select the top k keyphrases based on their weight, or top $N/3$ vertices based on their weight.

An advantage is that TextRank is able to extract phrases that do not occur often but might still be relevant. Furthermore, many graph-based approaches only rely on the document itself to extract keyphrases. In contrast, methods like rely on a large domain-specific corpus to determine the appropriate importance of each term.

4.2.4 PositionRank

The earlier described methods assume a uniform distribution over the words in the text because each word was initialized with a weight of 1. However, the position of the word in a document could also play a vital importance, but TextRank disregards this. Therefore, Florescu and Caragea (2017) introduced PositionRank. PositionRank differs from TextRank in two important elements.

Firstly similar to TextRank a word-graph $G = (V, E)$ is made, where each vertex represents a unique word that passed the syntactic filter. Thereafter, the vertices v_i and v_j are connected through an edge $(v_i, v_j) \in E$ if these words appear in the same co-occurance window, as described in 4.2.3; however, unlike TextRank the edges are weighted. The weight of an edge $(v_i, v_j) \in E$ is calculated by the co-occurrence count of the two words such that two vertex that co-occur often have a higher edge weight.

Secondly, the initial weights are biased towards words that appear at the beginning of the document and occur often. For example, in the TextRank algorithm a word that appears in the 3$^{\text{rd}}$, 6$^{\text{th}}$ and 12$^{\text{th}}$ position would simply have an initial weight of 1. In comparison, in PositionRank the sum of the initial weights is $\sum_{i=1}^{N} S(v_i) = 1$. To calculate the initial weight for each vertex two steps are required.

Firstly, in PositionRank the word position influences the weight, such that a word that appears in the 3$^{\text{rd}}$, 6$^{\text{th}}$ and 12$^{\text{th}}$ position has a weight of: $1/3 + 1/6 + 1/12 = 7/12$=0.583.

Secondly, we normalize the weight of each word, so that the sum of the weight is 1. A vector of Pagerank scores $\vec{S} = (S_i)_{0<=i<=N}$ is created, which is given in Equation 4.3.

$$\vec{S} = \left[\frac{S(v_1)}{S(v_1) + S(v_2) + \ldots + S(v_N)}, \frac{S(v_2)}{S(v_1) + S(v_2) + \ldots + S(v_N)}, \ldots, \frac{S(v_N)}{S(v_1) + S(v_2) + \ldots + S(v_N)} \right] \tag{4.3}$$

After initializing the weight vector, we can iteratively calculate the PageRank score for each vertex v_i with Equation 4.4. Note that unlike the original TextRank algorithm, the edges in PositionRank are weighted, such that two vertices v_i and v_j have a weighted edge w_{ij}. The weight of the edges is based on the co-occurrence count of the two words within a window of w successive tokens in the document.

$$S\left(v_i; t+1\right) = (1-d) \cdot S_i + d \cdot \sum_{v_j \in Adj(v_i)} \frac{w_{ji}}{\sum_{v_k \in Adj(v_j)} w_{jk}} S\left(v_j; t\right) \tag{4.4}$$

The final steps of the algorithm are the same as TextRank. More specifically, we define an error-rate epsilon, which means we assume convergence when $|S(v_i; t+1) - S(v_i; t)| < \epsilon$, for each $v_i \in V_i$. Next, words are sorted in descending order based on their PageRank weight. Afterward, words are combined and their weights are normalized if they are both in the top keyphrases k and are adjacent in the document.

4.2.5 Multipartite Graph

In many regards, the approach taken by Boudin (2018) is similar to the previously discussed graph-based algorithms. Namely, first a word-graph is constructed, after which a ranking algorithm is applied to assign a relevance score to each keyphrase. Furthermore, similar to PositionRank the position of a phrase in the document also influences its ranking.

However, as mentioned in Section 3.3.2, this approach focuses on ensuring topical diversity in the final set of keyphrases. This is done by forming a multipartite graph, where each topic consists of vertices belonging to a particular topic. In a multipartite graph vertices belonging to an independent set cannot be linked to each other, but they can be linked to vertices from another independent set. In our example, an independent set consists of a specific topic, where each vertex is a keyphrase candidate. Topics are formed based on the stem forms of the words they share using hierarchical agglomerative clustering with average linkage.

Edges between vertices are weighted according to the sum of the inverse distances between the candidate keyphrases in the document. As can be observed from Equation 4.5, recall that v_i represents vertex i. Recall that w_{ij} is the edge weight between these vertices. Note that $pos(v_i)$ denotes all the position of vertex v_i in the document

$$w_{ij} = \sum_{p_i \in pos(v_i)} \sum_{p_j \in pos(v_j)} \frac{1}{|p_i - p_j|} \tag{4.5}$$

Also, candidate keyphrases belonging to the beginning of the document are promoted by other candidates belonging to the same topic. Therefore, the edge weight between two vertices is adjusted through Equation 4.6. T_j is the set of candidates belonging to the same topic as vertex v_j. p_i in this case is the first occurrence of vertex v_i, and α is a hyperparameter, which controls the strength of the weight adjustment.

$$w_{ij} = w_{ij} + \alpha \cdot e^{\left(\frac{1}{p_i}\right)} \cdot \sum_{v_k \in \mathcal{T}(v_j) \setminus \{v_j\}} w_{ki} \tag{4.6}$$

In short, the last steps of the algorithms are similar to the other graph-based approaches; the ranking algorithm in Equation 4.4 is performed until the algorithm converges, that is a certain threshold ϵ is achieved. Lastly, candidate phrases are sorted in descending order based on their PageRank weight. Afterward, the phrases are combined and their weights are normalized if they are both in the top keyphrases and are adjacent in d.

4.3 Ensemble approaches

4.3.1 Ranking ensemble

To put it simply, the ranking ensemble selects the top keyphrases of each individual algorithms. Suppose we want to select the top k keyphrases for a single document. Let us denote $K_{ij}, ..., K_{ij}$ as the jth ordered keyphrases from algorithm i algorithm, and R_j are the jth ordered keyphrases of the ranking ensemble. It should be stressed that, with ordered keyphrases we mean the keyphrases are ordered from most to least relevant, based on their weight.

The algorithms starts by choosing K_{11} for the first keyphrase of the ranking ensemble, such that $A_{11} = R_1$. Afterwards, we choose K_{21} for the second keyphrase of the ranking ensemble *except* if $K_{21} = R_1$. In the scenario that $K_{21} = R_1$, we continue the process, so we choose K_{31} for the second keyphrase of the ranking ensemble such that $K_{31} = R_2$, except if $A_{31} = R_2$. We continue this process until we found the top k keyphrases for our ranking ensemble.

Of course, it matters which algorithms we start with because it will influence the order of our final set of keyphrases, which could influence our performance on ranking metrics (see Section 5.4.1). Therefore, in Section 5.5.2 we will explain how we selected the appropriate order of the algorithms.

Algorithm 1 ranking ensemble

input : top k ranked keyphrases of several algorithms
output: top k ranked keyphrases of the ranking ensemble
$j = 1$
for *i in range (1,k+1)* **do**
  if $R_i != K_{ij}$ **then**
    $R_i = K_{ij}$
  end
  else
    $R_i = K_{i(j+1)}$
  end
end

4.3.2 Voting ensemble

This is done by letting each individual algorithm vote for the final keyphrases. More specifically, if we want to extract k keyphrases for a single document, we first extract the k keyphrases of each individual algorithms. Thereafter, each unique keyphrase obtains a score. The score is based on the number of votes it obtained from the individual algorithms. The higher the number of votes, the higher the keyphrase is ranked in the final set k.

Chapter 5

Experimental setup

This chapter is devoted to explaining the experimental setup, and answering the first two sub-questions of this research. Section 5.1 gives a brief description of the data used in this research. Section 5.2 focuses on the setup for the annotation process. The various IAA for this research are explained in Section 5.3. Section 5.4 explains the evaluation methods for this research. The first two-sub questions of this research are also answered in this section. Lastly, Section 5.5 explains the implementation details of this research.

5.1 Data of this research

A description of Vilans is given in Section 2.1, where it is explained that Vilans focuses on long-term care. In general, their websites focus on long-term care, but each website has different specializations within this domain. For reproducibility purposes the data-preprocessing steps can be found in Appendix B. For an actual analysis of the data used in this research we refer to Appendix C.

Most importantly, in Section 2.3 we explained that the fifth sub-question of this research focuses on determining if the algorithms perform better for shorter or longer documents. Therefore, we would like to have an equal distribution of short and long documents in our final test set, so we can answer this sub-question. Fortunately, from the data analysis it becomes clear that Vilans has a combination of short and long documents. The next section explains how we used this information to organize the annotation process and create the validation and test set for the algorithms.

5.2 Annotation process

Reasons for using a gold standard

There are two main reasons we decided to create a gold standard by using human annotators. Firstly, the alternative is human evaluation, but as mentioned in

Section 3.5 comparing the results of different keyphrase extraction algorithms with human evaluation is time-consuming and expensive (Hasan and Ng, 2014). The reason being is that every document and algorithm has a unique set of keyphrases, which humans would need to evaluate. In contrast, by having a gold standard we can simply measure the performance of each individual algorithm by comparing the keyphrases of the algorithm with the gold standard. Secondly, most researchers use automated evaluation metrics, which makes it easier to compare our results with other research in this field.

5.2.1 Creating the gold standard

For this research, manual annotation was inevitable; this section describes the experimental setup to gather the annotations. Our experimental set up is based on the suggestions by Eisenstein (2018).

Three employees of Vilans were used to gather the annotations. Furthermore, domain-experts were chosen because the documents require much domain knowledge on long-term care. Initially, we started with a pilot version of 19 documents and two employees of Vilans. The reason for staring with a pilot version was twofold: 1) to avoid ambiguity among the annotators during the next stage of the process; 2) to obtain feedback regarding the task description and annotation guidelines.

In this paragraph the annotation guidelines are described, however, note that everything that is italicized was contained in the final task description, but *not* in the pilot version. The following paragraph explains why we slightly altered the original task description. In brief, the annotation guidelines mentioned the following:

1. The goal of a keyphrase is to shortly summarize the document. Keyphrases can be single words, but also a combination of coherent words. Examples are (1) elderly care; (2) long term care; (3) the Dutch Healthcare Authority; (4) healthcare.

2. In essence, everything can be a keyphrase. For instance, an organization, a person, the subject of the document etc.

3. The keyphrases should be distinguished in two categories: keyphrases contained in the document (extractive phrases), *and keyphrases which are absent in the document (abstractive phrases).* For keyphrases in the text the phrase can be anywhere in the document (including the title), *but for phrases outside of the text, annotators are requested to use their own creativity and domain knowledge.*

4. As a guideline, each document should contain about five keyphrases, *but deviation from this guideline is possible if the document requires less or more phrases for representative summarization. In addition, the keyphrases should be ordered in relevance; thus, from most relevant to least relevant.*

5. Annotators are requested to write down the abbreviated form of a word, if the abbreviated form appears in the document.

Table 5.1: The pilot documents used for the pilot version, which was also used as our validation set

type of documents	number of documents	number of words
short and long documents	19	118-1139

In short, there were three changes in the task description of the pilot and the final version. Firstly, in the pilot version, the idea was to only allow extractive keyphrases in the gold standard. This confined the task, but we assumed evaluating abstractive keyphrases with a gold standard would be too difficult. This can be contributed to the variations between semantically similar phrases, which are *not* the same according to the exact match. Secondly, in the pilot version, we decided to restrict the number of keyphrases of the annotators to five keyphrases. This setup was chosen because it would allow us to easily match the top five keyphrases of our system with the gold standard. In contrast, allowing a variable number of keyphrases by the annotators can lead to evaluation problems, which is discussed in 7.2.

Nevertheless, we obtained some feedback regarding both decision, so decided to slightly change the task description. More specifically, annotators were allowed to annotate abstractive keyphrases, but had to do so in a different column. This would allow us to compare the performance on the extractive keyphrases, and the combined set of extractive and abstractive keyphrases. Furthermore, during the second phase, we allowed annotators to annotate a variable number of keyphrases for each document. The reason for this decision was that annotators thought the limit of five keyphrases was too strict. As a result, they often had to remove certain keyphrases, or add phrases to fulfill this requirement.

For the final version, a total of three domain-experts were given a new set of documents to annotate: a total of 100 documents, which varied in length. More specifically, to answer *SQ5* of this research we need to have an even distribution between shorter and longer documents. The definition of what constitutes a short document is subjective, but we consider these to be all the documents that are shorter or equal to the length of the median of the corpus (341 words). On the contrary, longer documents are considered to have more than 341 words.

For the annotation process, the short and long documents are distributed according to a 50/50 split; that is to say 50 short and 50 long documents. It should also be mentioned that extreme outliers are omitted. We consider outliers to be documents belonging below the 10th percentile (less than 118 words), and longer than the 90th percentile (more than 1016 words). We assume that documents below the 10th percentile are not interesting to analyze because the reader can quickly read these without any keyphrases. Our pilot version

did contain one document with more than 1016 words, but in hindsight, this was considered to be too time-consuming to annotate. Therefore, to prevent jeopardizing the scalability of the annotation process we decided to change this upper bound to 1016 words, which still allows us to answer *SQ5* of this research.

Table 5.2: The final test set

type of documents	number of documents	number of words
short documents	50	118-341
long documents	50	342-1016
final test set	100	118-1016

To our knowledge, there are no external tools with regards to the annotation process in keyphrase extraction. Therefore, we provided the annotators with an Excel file, where the annotators could write down the keyphrases in two separate columns. The first column could be used for annotating extractive keyphrases, and the second column for annotating abstractive keyphrases. Also, the Excel file contained an external link to the URL of the document, such that the annotators were provided with all relevant context of the document (e.g., pictures).

5.2.2 Changes to the gold standard

After receiving the annotated documents, we verified if the annotators annotated the documents in accordance with the instructions. In brief, the final annotations obtained 1370 extractive keyphrases, but 7.13% of the keyphrases were not in accordance with the instructions. More specifically, 2.91% of the extractive keyphrases contained typographical errors, 3.86% annotations should have been put in the abstractive keyphrases column, and for 0.36% of the annotations the annotators clicked on the wrong URL of the document (annotating keyphrases belonging to another document). We manually changed the typographical errors to the original phrase in the document. Furthermore, there were 3.72 % data inconsistencies; that is phrases that were present in the document on the website, but not in the data set. Appendix D describes how we prevented that the annotation errors and data inconsistencies would incorrectly penalize the algorithms during the evaluation, After these changes, we normalized the annotated keyphrases, and the keyphrases of the algorithms. The process for this is explained in the next section.

5.2.3 Normalization of the keyphrases

The exact match is strict, so it is necessary to normalize all the keyphrases before calculating the IAA and evaluation. We provide the normalization steps we performed and an argumentation for each step:

- Lowercasing the words of the annotated keyphrases. The words of the algorithms were already lowercased, but the annotated keyphrases were not always lowercased. Therefore, we decided to lowercase the letters. As a result, keyphrases like "Langdurige zorg" would be changed to "langdurige zorg".
- Removing diacritics and accents to avoid mismatches between "cliënt" and "client".
- Removing stopwords in the annotated keyphrases since stopwords were not always annotated. This would change the annotated keyphrase: "dementiezorg voor elkaar" to "dementiezorg elkaar". To remove the stopwords we used the dutch stopwords of NLTK (Loper and Bird, 2009).
- Removing other characters that provide little to no semantic value but could lead to evaluation problems. We deleted these characters by using regular expressions. The characters we deleted were: exclamation marks, question marks, parentheses, quotation marks, and comma splices.
- Stemming the keyphrases as is often done in the literature to minimize the number of semantical mismatches (Hasan and Ng, 2014). To stem the keyphrases we used the Dutch Snowball stemmer from NLTK (Loper and Bird, 2009).

5.3 Measuring inter-annotator agreement

In general, to determine the granularity for keyphrases there are two commonly used agreement measures: 1) measuring the overlaps between annotators (i.e., percentage of agreement); 2) comparing annotations between annotators with a chance correction.

The first option can be insightful but should be used with great caution. Most notably, in pure categorization tasks like text classification the percentage of agreement can be misleading. In short, due to the restricted number of classes annotators could potentially have a high IAA by pure chance. A dissection of the exact reason is beyond the scope of this thesis, but we refer to the work of (Mathet et al., 2015, p. 441-443) for this explanation.

In the event that the NLP task is *not* a pure categorization task we argue that percentage agreement can be quite insightful. Specifically, we argue that keyphrase extraction belongs to the class of problems, which cannot be considered as a pure categorization task. We hypothesize that the agreement by chance in keyphrase extraction is low due to the possible number of keyphrases, the possible word combinations, and the variable number of keyphrases. Therefore, we utilized both IAA measures and compared their results.

However, it is worth mentioning that the gamma method (discussed in Section 5.3.3) can only be applied to extractive keyphrases since the substring of the keyphrases needs to be present in the original text. On the contrary, the

Jaccard index (as discussed in Section 5.3.1) can be applied to calculate the IAA for the extractive and abstractive keyphrases.

5.3.1 Number of relevant keyphrases per document

The Jaccard index and gamma method (discussed in Section 5.3.3) are not only influenced by the keyphrases the annotators wrote down, but also by the *number* of keyphrases they wrote down. Namely, if these are not equal to each other there is disagreement about how much phrases are necessary for effective summarization. Therefore, it is also interesting to investigate to what extent the annotators agree on the number of keyphrases that are necessary for summarization.

Table 5.3 illustrates that the number of keyphrases that are relevant for a document vastly differs per annotator. From these observations, we can observe there is a slight disagreement regarding the number of keyphrases between the first and second annotator. Interestingly, the third annotator seems to write down less extractive keyphrases, but more abstractive keyphrases. To determine if these variations are statistically significant we applied a Wilcoxon signed-rank test with the null hypothesis that the annotated keyphrases for both annotators are drawn from the same distribution. This statistical test was chosen as it does not assume a specific distribution for the two samples.

Moreover, we apply a significance level of $\alpha = 0.05$, but we apply six different tests, so the likelihood of incorrectly rejecting a null hypothesis (making a Type I error) increases (Mittelhammer et al., 2000). More specifically, if we apply six tests we can calculate the type I error as follows:

$$\begin{aligned} P(\text{ type I error }) &= 1 - P(\text{ no significant results }) \\ &= 1 - (1 - 0.05)^6 \\ &\approx 0.26 \end{aligned}$$

Therefore, we apply a Bonferroni correction (Bonferroni, 1936), so we achieve a significance of $\alpha/n = 0.05/6 = 0.0083$, where n denotes the number of tests.

Table 5.3: The average number of keyphrases each annotator wrote down

annotators	#extractive keyphrases	#abstractive keyphrases
annotator 1	4.83	0.33
annotator 2	4.46	0.74
annotator 3	3.81	1.33

From Table 5.4 and 5.5 we can observe that in five of the six tests the null hypothesis is rejected by low p-values ($\leq$0.0002). Thus, we can safely assume that for five of the six tested cases our samples are from different distributions, so we have sufficient evidence to assume that the number of keyphrases that are

Table 5.4: The results of the Wilcoxon signed-rank test for the number of *extractive keyphrases*

annotators	p-value	conclusion
annotator 1 and 2	0.03	do not reject H_0
annotator 1 and 3	4.29e-10	reject H_0
annotator 2 and 3	0.0002	reject H_0

Table 5.5: The results of the Wilcoxon signed-rank test for the number of *abstractive keyphrases*

annotators	p-value	conclusion
annotator 1 and 2	0.0002	reject H_0
annotator 1 and 3	3.37e-11	reject H_0
annotator 2 and 3	9.47e-05	reject H_0

considered to be relevant for each annotator statistically differs in five out of six events.

5.3.2 Inter-annotator agreement without chance correction

Consider the following scenario, where G_i consists of the keyphrases given by annotator i for a single document. In our experiments $i \in \{1, 2, 3\}$ because there are three annotators. To measure the overlap between keyphrases for each annotator pair, we can calculate the Jaccard Index, which measures the similarity between two finite sets. The formula for calculating the Jaccard index between two annotators i and j is given in Equation 5.1. It should be noted that $|G_i \cap G_j|$ is the cardinality of the intersection between G_i, and G_j, and $|G_i \cup G_j|$ is the cardinality of the union of G_i and G_j. Equation 5.1 shows how we can measure the Jaccard index for a single document.

$$J(G_i, G_j) = \frac{|G_i \cap G_j|}{|G_i \cup G_j|} \tag{5.1}$$

Note that $0 \leq J(G_i, G_j) \leq 1$, where 0 means there is no overlap between the keyphrases and 1 means there is a complete overlap between the keyphrases. It should be stressed that a Jaccard index of 1 is only achieved if and only if the keyphrases are the same according to the exact match, and both annotators annotated the same number of keyphrases for each document. To calculate the similarity for each pair of annotators over all documents, we simply averaged the Jaccard indexes for each pair.

The Jaccard indexes between the annotators are given in Table 5.6. From this table, we can observe there is a percentage agreement of 23% between

annotator 1 and 2, 25% between annotator 1 and 3, and 23% between annotator 2 and 3. If we average these Jaccard indices we obtain an overall similarity of 24%. It is noteworthy, to mention that all the values of the Jaccard index are similar; suggesting that the annotators usually agree on one or two keyphrases, but disagree on the rest of the keyphrases.

Table 5.6: The similarity (rounded to two decimals) between the *extractive* keyphrases based on the Jaccard index.

annotators	similarity
annotator 1 (G_1) and 2 (G_2)	0.23
annotator 1 (G_1) and 3 (G_3)	0.25
annotator 2 (G_2) and 3 (G_3)	0.24
average	0.24

From Table 5.7 we can clearly see that there is little or no similarity between the *abstractive keyphrases* between the annotators. This is rather unsurprising because there were no strict guidelines for these keyphrases: annotators were allowed to use their own creativity. However, from this table we can conclude that implementing a gold standard limits us with regards to evaluating abstractive keyphrases. Namely, there is so much variability in the human language, so evaluating abstractive keyphrases with a gold standard most likely does *not* accurately show the true quality of the abstractive keyphrases.

Table 5.7: The similarity between the *abstractive* keyphrases based on the Jaccard index.

annotators	similarity
annotator 1 (G_1) and 2 (G_2)	0.0
annotator 1 (G_1) and 3 (G_3)	0.0
annotator 2 (G_2) and 3 (G_3)	0.02
average	0.006

The Jaccard index is an intuitive and simple approach, which effectively gives an indication of the agreement measure; however, there is no chance correction for extractive keyphrases. To combat this pitfall, we propose a more sophisticated approach in Section 5.3.3, which is derived from the work of Mathet et al. (2015).

5.3.3 Inter-annotator agreement with chance correction

The method suggested by Mathet et al. (2015) is called the unified and holistic method gamma. In essence, it effectively combats the problems with traditional categorization agreement measures proposed in Section 3.6.1. The goal of this

section is to explain the gamma method and mention the modifications we made to make it relevant for keyphrase extraction. The notation used for this section can be found in Table 5.8. Note that $\gamma = 1$ corresponds to a perfect agreement, and $\gamma < 0$ corresponds to an agreement less than pure chance.

Notation

Table 5.8: Formal notation used in this subsection

notation	explanation
γ	inter-annotator agreement
c	corpus of available annotation
$\delta(d)$	disorder of document d
$\delta_e(d)$	expected disorder of document d
$cat(i)$	category assigned to unit i
$start(i)$	start boundary of unit i
$end(i)$	end boundary of unit i
$\bar{x} = \sum_{k=1}^{n} x_k/n$	average number of annotations per annotator
$\forall k \in [1, n]$,	let x_k be the number of units by annotator a_k for d
$a = a_k, \forall z \in [1, x_z]$	u_k^a unit from a of rank z
$\mathcal{U}$	set of units from all annotators

Intuition behind gamma

Figure 5.1 gives three possible scenarios to explain the intuition behind calculating the agreement. The annotations on the left correspond to the best possible agreement because the annotators agree on the category and boundary of each unit. As a result, $\gamma = 1$. In contrast, the annotations on the right are the worst possible agreement because the annotators do not agree on the category and the boundaries. In other words, the agreement is less than chance, such that $\gamma < 0$.

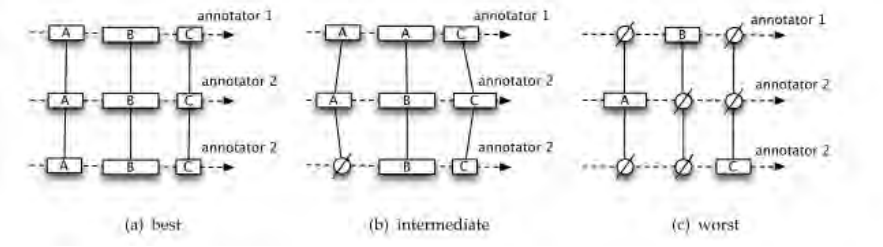


Figure 5.1: Examples of possible disorders. Source: (Mathet et al., 2015, p. 454)

Thus, the idea of gamma is to compute the disorder between unitary alignments. Afterward, the global disorder can be computed by averaging the unitary

disorders.

Calculating the unitary disorder

Each unit has a category and location given by its start and end boundaries in the text. Mathet et al. (2015) argues that equality between two units is a function of three variables: category, start boundary, and end boundary.

However, for keyphrase extraction there are only two categories: relevant and non-relevant. In keyphrase extraction each unit consists of a keyphrase, and can therefore be considered to belong to the category relevant. Recall that $cat(i)$ denotes the category of unit i, and $cat(j)$ denotes the category of unit j. Therefore, for keyphrase extraction we define $cat(i) = \text{cat}(j)\ \forall (i,j) \in \mathcal{U}^2$. As a result, we slighly adjust the original definition of equality by Mathet et al. (2015). This results in the following definition:

$$\forall (i,j) \in \mathcal{U}^2, i = j \Leftrightarrow \begin{cases} \text{start}(i) = \text{start}(j) \\ \text{end}(i) = \text{end}(j) \end{cases} \tag{5.2}$$

In our implementation, the start boundary corresponds to the first character of the keyphrase in d, and the end boundary corresponds to the last character of the keyphrase in d. In our implementation, these characters were matched using regular expressions.

We define dissimilarities according to Equation 5.3. We can observe that the distance between two unitary alignments is defined as 0 if $i = j$, so the start and end position should align. Also note that the distance between i and j is symmetric, such that $dis(i,j) = dis(j,i)$.

$$\forall (i,j) \in \mathcal{U}^2, \begin{cases} dis(i,j) = dis(j,i)\ \text{(symmetric relationship)} \\ i = j \Rightarrow dis(i,j) = 0 \end{cases} \tag{5.3}$$

However, we need an actual formula to calculate the distance between two units, which we can then use to measure the dissimilarity. The formula in Equation 5.3 mentioned that $dis(i,j) = 0$, if $i = j$, but it also introduced the notion of inequality between the two units because annotators often disagree. Inequality is defined if two units are dissimilar by start and end position, but the magnitude of the disorder is dependent on how much the start and end boundaries of two annotations differ.

Therefore, to calculate the actual distance between two unitary alignments we introduce two concepts: 1) a cost function; 2) a formula for calculating the distance.

The cost is utilized, if for unit $u_k^{a_1}$ of annotator a_1, we cannot align it with a unit $u_k^{a_2}$ of annotator a_2. We adhere to the recommendations of Mathet et al. (2015), and set this cost function to 1 in our implementation.

$$\begin{aligned} \forall i \in \mathcal{U}, dis\,(i, u_0) = dis\,(i_\emptyset, i) = \Delta_0 \\ \text{and } dis\,(i_0, i_0) = \Delta_0 \end{aligned} \tag{5.4}$$

The formula for the distance is given in Equation 5.5. In the numerator we sum up the units of the start and end boundaries, and in the denominator we ensure this dissimilarity is scale-dependent by summing up the length of both units. The values are squared to give a penalty to dissimilarities that are large.

$$dis(i,j) = \left(\frac{|\,\text{start}(i) - \text{start}(j)| + |end(i) - \text{end}(j)|}{(end(i) - \text{start}(i)) + (\text{end}(j) - \text{start}(j))} \right)^2 \cdot \Delta_{\emptyset} \tag{5.5}$$

The disorder between each keyphrase alignment can be defined by $\delta(\breve{a})$. From Equation 5.6 we can observe the results are averaged to make it independent of the number of annotators available.

$$\breve{\delta}(\breve{a}) = \frac{1}{C_n^2} \cdot \sum_{(i,j) \in \tilde{a}^2} dis(i,j) \tag{5.6}$$

Afterwards, the global disorder denoted by $\bar{\delta}(\bar{a})$ can be calculated. We sum up the individual keyphrase alignment divided by the mean number of keyphrases per annotator. The average value is computed, such that the disorder is not influenced by the length of the document.

$$\bar{\delta}(\bar{a}) = \frac{1}{\bar{x}} \cdot \sum_{k=1}^{|\bar{a}|} \breve{\delta}\,(\breve{a}_k) \tag{5.7}$$

Lastly, the expected disorder (agreement by chance) is obtained by randomly shuffling the unit of keyphrases. An example of this can be seen in Figure 5.2.

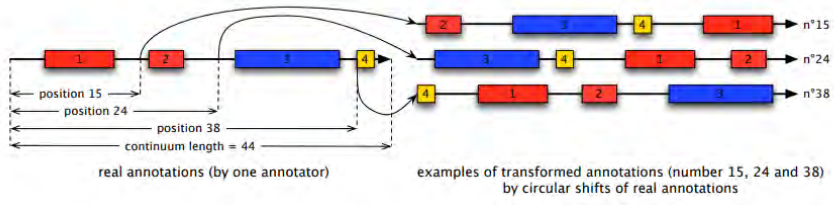


Figure 5.2: Examples of possible disorders. Source: (Mathet et al., 2015, p. 459)

To conclude, the formula for the gamma method is given in Equation 5.8. Recall that $\delta(d)$ is the global disorder for document d, and $\delta_e(d)$ is the expected disorder of document d.

$$\forall d \in c, \gamma = 1 - \frac{\delta(d)}{\delta_e(d)} \tag{5.8}$$

We used a Java open-source package developed by the authors to implement the gamma method. [1] We calculated the gamma for each individual

[1] The full software package can be found at: http://gamma.greyc.fr

document, and averaged these results: $\sum_{i=1}^{c} \gamma_i / c$, where γ_i is the disorder of the ith document, and c is the number of available documents with annotations, which is 100 as described in Section 5.2. The Table in 5.9 shows the results of implementing the gamma method. The lowest value obtained was -0.06; an agreement slightly less than chance. However, it should be mentioned that an agreement less than chance occurred only for one document. The maximum value was 0.56, and the total average was 0.22. Interestingly enough, the average of the gamma is close to the value of the Jaccard indexes given in Equation 5.6. The next section interprets the findings of the two implemented IAA methods.

Table 5.9: The results of implementing the gamma method

gamma	average	min.	max.
γ	0.22	-0.06	0.57

5.3.4 Final remarks

The results of implementing the two different methods indicate that the agreement by chance is low. As can be seen in Table 5.10, the average of the gamma IAA is only 0.02 lower than the Jaccard index, which does not utilize a chance correction. Thus, these results suggest that our hypothesis is true: the agreement by chance in keyphrase extraction is small because the agreement *with* a chance correction is only slightly lower than *without* a chance correction. We assume the agreement by chance is low due to the possible number of keyphrases, the possible word combinations, and the variable number of keyphrases.

Furthermore, the results indicate a low agreement among the annotators, which suggests that keyphrase extraction is subjective. This might explain the reason why keyphrase extraction performs worse than many other NLP tasks. In general, NLP tasks that are considered less subjective also have a higher IAA. As a result, tasks which are more objective obtain a higher accuracy than tasks that are subjective. For example, in the English language, the inter-annotator agreement of part-of-speech tagging is often around 97% (Manning, 2011). Consequently, many current state-of-the-art part-of-speech taggers obtain an accuracy of almost 100% (Qi et al., 2020) when tagging material from the same source and epoch on which they were trained (Manning, 2011).

Another important observation from Table 5.10 is that there is little agreement between the abstractive keyphrases. This is illustrated by the fact that the Jaccard index of the abstractive keyphrases (0.006) is significantly lower than the extractive keyphrases (0.24). We strongly believe this is due to the fact that there are many possible abstractive keyphrases that are semantically similar but are *not* considered similar due to the exact match. These results suggest that human evaluation (as discussed in Section 3.5) is a better method for evaluating abstractive keyphrases.

Table 5.10: The results of implementing both IAA metrics

method	average
Jaccard index (extractive)	0.24
γ (extractive)	0.22
Jaccard index (abstractive)	0.006

5.4 Evaluation

5.4.1 Evaluation metrics for this research

For this research, common automated evaluation metrics are calculated to determine the performance of the algorithms. A common evaluation metric for classification tasks is accuracy, which measures how often the classifier is correct. However, it should be stressed that utilizing accuracy could be misleading in the case of class-imbalances. To illustrate this problem consider we would have a document where 2.0% of the phrases are considered keyphrases according to the gold standard. A classifier predicting only non-keyphrases, would already yield an accuracy of 98% in this scenario.

There are alternative evaluation metrics, which are better at handling class-imbalances. These are precision, recall, and F_1 score and are widely used in the most recent keyphrase extraction literature (Florescu and Caragea, 2017; Meng et al., 2017; Yuan et al., 2018; Zhang et al., 2016). To understand exactly how accuracy, recall, precision, and F_1 score are calculated we need to introduce a term called the confusion matrix, which is given in Table 5.11. From this table we can conclude the following:

- **True positive (TP)**
  The algorithm correctly classifies the phrase as a keyphrase.
- **False Negative (FN)**
  The algorithm incorrectly classified the phrase as a non-keyphrase.
- **False Positive (FP)**
  The algorithm incorrectly classified the phrase as a keyphrase.
- **True Negative (TN)**
  The algorithm correctly classified the phrases as a non-keyphrase.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5.9}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.10}$$

$$Precision = \frac{TP}{TP + FP} \tag{5.11}$$

Table 5.11: Confusion matrix of the predictions

		gold standard: positive	gold standard: negative	total
predicted keyphrases	positive	TP	FP	$TP + FP$
	negative	FN	TN	$FN + TN$
	total	$TP + FN$	$FN + TN$	$TP + FP + FN + TN$

$$F_1 score = \frac{2 * P * R}{P + R} \tag{5.12}$$

Thus, from Equation 5.9 we can see that accuracy is unable to deal with class-imbalances because it results in many true negatives.In comparison, Equation 5.10 and Equation 5.11 show that recall and precision do not take true negatives into account.

Specifically, recall shows how many relevant keyphrases are selected out of all the relevant keyphrases, whereas precision shows how many of the selected keyphrases are actually relevant. Ideally, one would want to maximize the precision and recall of a system, but in most practical situations this is infeasible. A measure that combines both precision and recall is called the F_1 score, which is given in Equation 5.12.

Another metric used in keyphrase extraction is MAP (Boudin, 2018). Regular precision calculates how many of the keyphrases of the algorithm are relevant according to the gold standard, but precision is unable to take the ranking of the keyphrases into account. On the contrary, MAP does take the ranking of the keyphrases into account, such that an algorithm is evaluated on the number of keyphrases of the algorithm that are relevant according to the gold standard, and *the order* of the keyphrases of the algorithm.

We define $AP(d)$ as the average precision (AP) for a single document d. For each document, we extract a fixed number of keyphrases k according to a specific algorithm. Furthermore, Let $P(d, k)$ be the precision for document d when only the first k keyphrases are considered.

$$AP(d) = \frac{\sum_{k=1}^{k} P(d, k)}{k} \tag{5.13}$$

After calculating the AP for each individual document, the MAP can be calculated by averaging the AP over all the documents. We define n to be the total number of documents we are evaluating. This gives us the following equation:

$$MAP = \frac{1}{n} \cdot \sum_{d=1}^{n} AP(d) \tag{5.14}$$

5.4.2 Evaluation setup with multiple annotators

This section focuses on answering the first two sub-questions of this research. Recall the first sub-question of this research:

SQ1: What can be considered the gold standard of keyphrases for each document, and how can we determine the granularity of these keyphrases?

It is worth mentioning that several options arise when we evaluate the algorithms on annotations given by multiple human annotators. One option is to combine the annotations into one set consisting of all the unique gold standard keyphrases. The major advantage of this approach is the increased robustness when measuring precision (Sterckx et al., 2018). Namely, annotators tend to write down similar keyphrases in semantically different ways, which are hard to distinguish due to the exact match restriction.

The only drawback of the first approach is that it leads to a large set of gold standard keyphrases, which does not allow us to accurately calculate the recall of the algorithms. An argumentation for this statement is given in the answer to the next sub-question. Therefore, the second option we propose is inspired by Sterckx et al. (2018), which compares the methods with the annotations given by an individual annotator. This effectively allows us to calculate the recall of the systems.

A benefit of the second option is that is also influenced by the number of annotators that annotated the keyphrase. For instance, consider the following scenario: all three annotators wrote down a certain keyphrase, but this is not in the top five ranked keyphrases by our algorithm. Thus, this is a major error of the algorithm, which should be reflected in the evaluation metric. However, the first option fails to deal with these errors, but the second option measures these errors because we average the recalls for each individual annotator. It should be mentioned that we explained in Section 5.3 how we determined the granularity of the gold standard.

SQ2: Which quantitative evaluation metrics are appropriate for the developed algorithms?

The previous sub-question provided argued that two gold standards are necessary for appropriate evaluation: one gold standard consisting of the combined unique keyphrases, and one of the individual gold standard keyphrases. For the combined unique keyphrases as the gold standard, we limited ourselves to the top five ranked keyphrases because this is a ranking problem. We provide three reasons for limiting ourselves to the top five keyphrases. Firstly, it was possible to extract more than five keyphrases per document, but we suspected that this would degrade the quality of the keyphrases. Secondly, it is likely that if these keyphrases are used for summarizing a document a reader may often not read more than the top five keyphrases. Therefore, it is much more interesting to calculate the precision for the top five ranked keyphrases than the top ten or

fifteen. Thirdly, as described in Section 5.2 we suggested the annotators to write down five keyphrases per document. Hence, this led us to use the precision@k and MAP@k, where k is set to five in our experiments.

For the second option, we calculated the recall@5 and F_1 score@5 for each annotator, and averaged these results. As explained in Section 5.2.1, we also instructed the annotators to annotate abstractive keyphrases. Therefore, we also determined the performance of these keyphrases. In the evaluation stage, our approach consisted of splitting the keyphrases into two sets: extractive, and extractive & abstractive keyphrases. It should be noted that the initial algorithms were unable to deal with abstractive keyphrases. For this reason, our algorithms were unable to achieve 100% recall on the combined set if the annotator wrote down an extractive keyphrase. Lastly, it should be stressed that we are measuring macro precision and recall. Thus, we assume that extracting relevant keyphrases is equally important for each document.

Section 5.3 proposed the idea that keyphrase extraction is a subjective task, so obtaining a high $F1$ score could be unattainable. Therefore, to give our experimental results more context we also decided to calculate the average mutual $F1$ score for the extractive keyphrases. For calculating the mutual $F1$ score we took the same approach as Kunneman et al. (2015). In short, we treat the keyphrases of one annotator as the keyphrases of the algorithm, and the keyphrases of another individual annotator as the gold standard. Repeating this procedure three times for the all combinations gives us three $F1$ scores, which allows us to calculate the average mutual $F1$ score. Consequently, we can determine how consistent our algorithms are with the annotators as they are among each other.

To summarize, we propose that to accurately determine the precision of the algorithms the first option is appropriate. Therefore, we decided to use this approach to calculate the precision@5, and MAP@5. Also, this allows us to determine if the quality of the keyphrases becomes worse if we extract more keyphrases, which would be the case if the MAP@5 is lower than the precision@5. For the second option we focus on determining the recall@5 and $F1$ score@5 on the extractive keyphrases, and extractive & abstractive keyphrases. Moreover, for effectively answering *SQ5* of this research it would be appropriate to compare the short and long document side by side. Therefore, for answering this sub-question we limit ourselves to the most important metrics for this problem: MAP@5, and $F1$ score@5 (for extractive keyphrases).

5.4.3 Model validation

The experiments described in Section 5.2 resulted in a test set of 100 documents; however, a test set of 100 is relatively small. Therefore, to obtain insight in the true parameter we are estimating with our experiments, we constructed confidence intervals for each algorithm. Recall that the true population parameter we are interested in are MAP@5 and precision@5 for the first option, and recall@5 and $F1$ score@5 for the second option. In addition, we applied statistical tests to determine if the results were statistically significant.

The confidence intervals were obtained through empirical bootstrapping (Davison and Hinkley, 1997). More specifically, the basic bootstrap (with replacement) was used, which is given in Equation 5.15. $\widehat{\theta}$ is the estimated population parameter (e.g., MAP@5), and $\theta^*_{(1-\alpha/2)}$ denotes the $1-\alpha/2$ percentile of the bootstrapped coefficients θ^*. For our experiments we used 10 000 bootstraps.

We set the significance level to the usual $\alpha = 0.05$, so we obtain a confidence interval of $(1 - 0.05)100\% = 95\%$. However, we want to emphasize that we are constructing seven confidence intervals for each evaluation metric, therefore, we should slightly adjust our significance level. Applying a standard Bonferroni correction on our significance level gives us a significance of $\alpha/m = {}^{0.05}/_{7} = 0.0071$. Therefore, we technically obtain a confidence interval of $(1 - \alpha/m)100\% = (1 - {}^{0.05}/_{7})100\% = 99.28\%$.

$$\left(2\widehat{\theta} - \theta^*_{(1-\alpha/2)}, 2\widehat{\theta} - \theta^*_{(\alpha/2)}\right) \tag{5.15}$$

Moreover, statistical tests were used to determine if the results were statistically significant. There are multiple statistical tests used in NLP research to validate the results of two algorithms on the same test set (Dror et al., 2018). Most notably, the paired t-test is often used; however, this is a parametric test, so both distributions should come from a normal distribution. Although this statistical test is widely used in many NLP research we prefer performing a randomization test. Namely, Cohen (1995) claims that the randomization test performs as well as the t-test when the normality assumption is met, but that the randomization test outperforms the t-test when the normality assumption is unmet. Experimental findings by other researchers also recommend using randomization tests (Dror et al., 2018; Hoeffding, 1952; Noreen, 1989; Smucker et al., 2007; Urbano et al., 2019).

Other popular non-parametric tests are the sign test and the Wilcoxon signed-rank test. However, the sign-test is not preferred because it only determines if algorithm A performed better or worse than algorithm B, but does not take the magnitude of these differences into account, which leads to more type II errors (Urbano et al., 2019). Alternatively, the Wilcoxon test can be used since this does take differences into account, but recent research shows it is prone to making many type I errors, especially at low significance levels. Especially as the sample size increases, they are overconfident and show a clear bias towards small p-values (Urbano et al., 2019).

All statistical tests considered the permutation test is the most suitable option for reasons previously discussed. Furthermore, it is worth mentioning that we only tested our results on statistical significance for the MAP@5 on the combined gold standard. We opted for this approach because of two reasons:

1. We argue that the MAP@5 on the combined gold standard is least susceptible to evaluation errors, for reasons discussed in Section 5.4.2. Furthermore, MAP@5 is preferred over precision@5 because it takes the order of the keyphrases into account.

2. If we were to test our results on all established evaluation metrics this would lead to many statistical tests. Therefore, increasing the probability of a type I error. We could have solved this by lowering the significance level with a Bonferroni correction with a higher m (number of statistical tests), but this would have increased the probability of a type II error.

5.5 Implementation details

For the statistical algorithms, we utilized the well-known Scikit-learn machine learning library (Pedregosa et al., 2011). For the graph-based algorithms we used the keyphrase extraction library of Boudin (2016). Both libraries are open-source, and can be implemented with the Python programming language.

5.5.1 Extracting domain-specific keyphrases

Description of the thesaurus

The thesaurus used is specific to the domain of healthcare, and was obtained by Vilans.[2] In total, there are 47 874 unique phrases in this thesaurus. All phrases have one preferred term (descriptor), and phrases can have several alternative versions (non-descriptors). The phrases are inter-connected by semantic relations of two types: links for related terms, and links for broader terms. For example, the Dutch phrase "diabetes" (descriptor) has a synonym relationship with "suikerziekte", a broader term relationship with "diabetes mellitus", and a related term relationship with "insuline". In total, there are 212 816 semantic relationships between the phrases.

Extracting abstractive keyphrases with a thesaurus

There are various approaches to extracting abstractive keyphrases with a thesaurus. However, it is worth mentioning that it can also lead to many false positives. For example, if one would simply extract all related or broader terms these could be actually irrelevant for the specific document. Therefore, we decided to take a rather cautious approach to limit the number of false positives. In brief, our approach consisted of two steps:

1. Convert all non-descriptors in the final set of keyphrases to descriptors. For instance, if "suikerziekte" (non-descriptor) is in the top k keyphrases we would replace it by its descriptor, which is "diabetes".

2. Extract broader terms, if and only if there are two keyphrases in the top X set of keyphrases from the algorithm that have the same phrase as the broader term. The exact definition of X is given in Section 6.3.1, after observing the experimental results. As an example, imagine we extract the keyphrases of a document with an algorithm, and find the keyphrases

[2] Neo4J (a graph database) was used for storing the thesaurus

"diabetes", and "maternale overerving" in X. Since "MIDD-type diabetes" is a broader term of both "diabetes", and "maternale overerving", the phrase "MIDD-type diabetes" is also extracted.

5.5.2 Parameter settings

As described in Section 5.2 we also performed a pilot program, which allowed us to use 19 documents as an evaluation set.

Our approach regarding hyperparameter tuning is slightly different than most comparative literature on unsupervised keyphrase extraction. Namely, most papers simply test all their hyperparameters on their final test set and usually conclude that the best-performing hyperparameters can be considered the best. Although this works for testing individual algorithms this would be not ideal in our scenario since we are using multiple ensemble methods, and their performance is dependent on the performance of the individual algorithms. For this reason we would like to know which algorithms and hyperparameters perform the best on the validation set. Considering the fact that this is a small validation set hyperparameter tuning was kept to a minimum; only the co-occurance window (w) of TextRank and PositionRank is optimized.

Based on previous research w can influence the performance of TextRank and PositionRank, which also influences the performance of our ensemble methods. Therefore, we determined the overall performance with $w \in \{2, 4, 6, 8, 10\}$ on our validation set. We found that $w = 4$ gave the best performance for TextRank, and $w = 6$ for PositionRank. These hyperparameters were evaluated on the MAP@5 and precision@5 of the combined gold standard. The other hyperparameters of the graph-based algorithms have proven to be optimal on datasets with different statistical properties (Boudin, 2018; Florescu and Caragea, 2017). Therefore, we decided to utilize these recommended hyperparameters. More specifically, for PositionRank and TextRank we decided to set error-rate (ϵ) to 0.0001, and the damping factor (d) to 0.85. For MultipartiteRank we set α to 1.1.

For all graph-based approaches, we limit our candidate words to noun phrases for reasons discussed in Section 4.2.3. Specifically, the part-of-speech of the word should be a noun, proper noun or adjective. Moreover, Section 4.2.3 mentioned that TextRank uses lexical units to calculate the co-occurrence window, but it should be mentioned that Mihalcea and Tarau (2004) did not mention which lexical units are taken into account for calculating this window. Therefore, we make the following assumptions for the graph-based approaches: sentence boundaries are not taken into account, and stopwords and punctuation marks are taken into account.

For the statistical models described in Section 4.1, the parameters were chosen based on recommendations from Scikit-learn. It should be mentioned that Sci-Kit learn allows us to change several parameters, three of which argue for our experiments: minimum document frequency (min. df), maximum document frequency (max. df), and n-gram range. The minimum document allows us to ignore terms that appear in less than a certain number of docu-

ments. We set this to the standard of 1 because new words are often created (e.g., Covid-19). Thus, setting a minimum threshold could miss these potential keyphrases. The maximum term frequency is also not restricted, but we did remove Dutch stopwords with the Natural Language Toolkit (Loper and Bird, 2009). In short, the only parameter we adjusted was the n-gram range, such that unigrams, bigrams, and trigrams would be considered.

To determine the inverse-document frequency of each term for we used 8 678 documents within the long-term care. For an elaborate description of these documents, we refer to Appendix C.

To summarize, the hyperparameter tuning, and determining the optimal order for the ranking ensemble was as follows:

1. Determine the optimal co-occurrence window of TextRank and PositionRank on the validation set.
2. Determine the best-performing algorithms on the validation set, so that this information can be used for the ranking ensemble

The hyperparameters for our experiments for the statistical algorithms can be seen in Table 5.12, Also, the hyperparameters for the graph-based algorithms can be seen in 5.13. Note that with "method" we refer to the method used for hierarchical clustering.

Table 5.12: Parameters used for the statistical algorithms on the test set.

algorithm	n-gram range	min. df	max. df
tf	1-3	1	1
tf-idf	1-3	1	1

Table 5.13: Parameters used for the graph-based algorithms on the test set.

algorithm	n-gram range	ϵ	w	d	α	threshold	method
textrank	1-3	0.0001	4	0.85	-	-	-
positionrank	1-3	0.0001	6	0.85	-	-	-
multipartiterank	1-3	0.0001	-	-	1.1	0.74	average

Table 5.14: Order the keyphrases were chosen for the ranking ensemble (based on results from the validation set).

best algorithms
tf-idf
PositionRank
TextRank
MultipartiteRank
term frequency

Chapter 6

Experimental results

This section is devoted to answering several of our research sub-questions by showing our experimental results, as defined in Chapter 5. First, we show the combined results on the overall test set and show the results of the statistical tests to determine if the results were statistically significant. However, as described in Section 2.1: Vilans has a combination of short and long documents. For this reason, it is important to have a robust algorithm that works for short and long documents. Therefore, we also present the results of the short and long documents.

6.1 Main results

This section focuses on answering the third and fourth sub-question of this research. We first show the results of option one as described in Section 5.4.2; so we present precision@5 and MAP@5 on the combined gold standard. Recall our third sub-question:

SQ3: **How accurately can unsupervised algorithms extract keyphrases, in comparison to the gold standard?**

Our results on the combined gold standard can be found in Table 6.1. From this table, we can deduce that the voting ensemble obtained the best results because 46.4% of the top five keyphrases are relevant. However, we can observe that the MAP@5 was significantly higher (58.2%). These results indicate that the quality of the keyphrases degrades if we extract more keyphrases. The second best algorithm is the ranking ensemble, which obtained a MAP@5 of 53.4%. The best-performing individual algorithm (tf-idf) obtained a MAP@5 of 50.7%. Surprisingly, even a simple technique like tf provided satisfactory results for the top five ranked keyphrases.

From Figure 6.1 we can observe that the voting ensemble performed better for all the top five ranks of the MAP. It is worth mentioning that the MAP gets significantly worse if we extract more keyphrases. For instance, the MAP@1

Table 6.1: Results of precision@5 and MAP@5 on the combined gold standard for all documents in the test set.

	extractive	
algorithms	precision@5%	MAP@5%
tf	41.4 (36.0, 47.8)	48.9 (41.7, 57.8)
tf-idf	39.4 (34.0, 46.2)	50.7 (43.8, 59.9)
TextRank	34.8 (29.6, 41.2)	43.9 (37.1, 53.0)
PositionRank	38.0 (32.6, 45.0)	49.2 (42.6, 57.4)
MultipartiteRank	37.8 (32.6, 43.8)	43.1 (36.7, 51.5)
ranking ensemble	41.9 (37.3, 46.5)	53.4 (47.4, 59.6)
voting ensemble	**46.4 (40.7, 52.1)**	**58.2 (51.5, 64.9)**

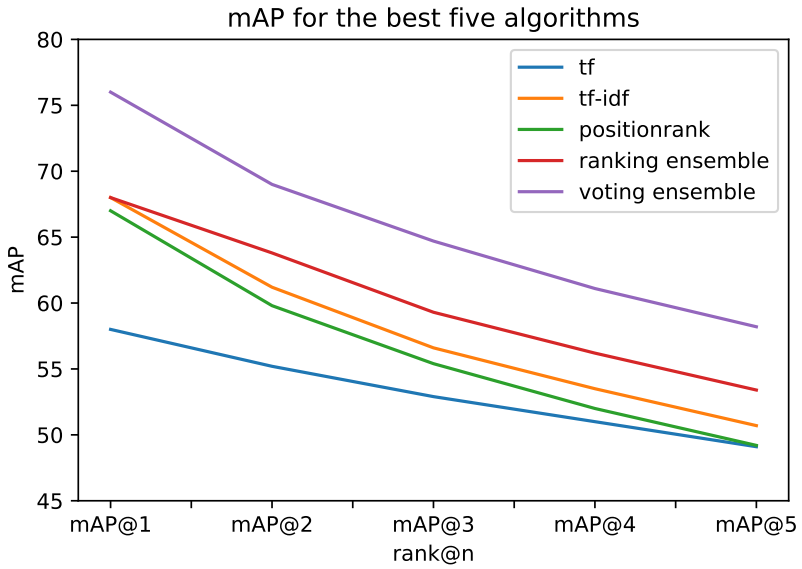


Figure 6.1: MAP for various ranks

(which is equal to the precision@1) for the voting ensemble was 76%, while the MAP@5 was 58.2%. Furthermore, it should be noted that the results of the individual algorithms also contradict with Table 6.1. More specifically, Table 6.1 suggest that a simple method like tf performed nearly as good as more complex methods such as tf-idf and PositionRank. However, Figure 6.1 illustrates that tf-idf and PositionRank performed much better for at least the top two ranked keyphrases, but the performance of these algorithms slowly converged when we extracted more keyphrases.

***SQ4:* Which algorithms obtain the best results on the established evaluation metrics? And can an ensemble method outperform each individual model?**

The results in the previous section illustrated two key findings. Firstly, 46.4% of the top five extracted keyphrases of the best-performing algorithm are relevant according to the combined gold standard. Secondly, the quality of the keyphrases degrades if we extract more keyphrases. Next, we show the results of the second set of evaluation metrics as described in Section 5.4.2; so we present recall@5 and $F1$ score@5 on the individual gold standard. From Table 6.2 we can observe that the voting ensemble outperformed the other algorithms in terms of recall@5 and F_1 score@5. Nonetheless, our results are significantly worse because we have fewer keyphrases for the gold standard. We can also observe that the F_1 score on the extractive & abstractive set is lower than the extractive set. However, this is to be expected because all algorithms are extractive, so they are unable to handle absent keyphrases. Nevertheless, the difference between the $F1$ scores is relatively small.

The annotators obtained an average mutual $F1$ score of 35.1% on the extractive keyphrases. Therefore, this suggests that there is still room for improvement for the algorithms.

Table 6.2: Results of the recall@5 and F_1 score@5 on the individual gold standard for all documents in the test set.

	extractive		extractive & abstractive
algorithms	recall@5 %	F1-score@5 %	F1-score@5%
tf	28.1 (24.0, 33.2)	25.7 (22.1, 30.5)	24.2 (20.7, 28.4)
tf-idf	27.1 (23.1, 32.1)	24.8 (21.1, 29.2)	23.3 (19.8, 27.6)
TextRank	23.4 (19.9, 28.1)	22.0 (18.5, 26.5)	20.6 (17.2, 24.7)
PositionRank	26.8 (22.4, 31.8)	24.5 (20.6, 29.2)	23.0 (19.2, 27.8)
MultipartiteRank	26.4 (22.7, 31.1)	24.3 (20.9, 28.5)	22.7 (19.5, 26.8)
ranking ensemble	29.9 (26.1, 33.6)	27.5 (24.0, 30.8)	25.6 (22.3, 28.8)
voting ensemble	**31.4 (27.4, 35.5)**	**28.8 (25.4, 32.3)**	**26.8 (23.6, 30.2)**

The previous section indicates that the voting ensemble provided the best results on all established evaluation metrics. To validate if the results were statistically significant we used the permutation test, as described in Section 5.4.3. Let $B_1, ..., B_{100}$ be the average precision of our baseline system on the documents of the test set. Moreover, let $E_1, ..., E_{100}$ be the average precision on the same documents, but for our voting ensemble. Furthermore, let $D_i = E_i - B_i$ be the difference in between the average precision for document i. $\bar{D}$ is the average of the differences for average precision. Thus, our null hypothesis is:

$$\begin{aligned} H_0 &: \bar{D} = 0 \\ H_1 &: \bar{D} \neq 0 \end{aligned} \tag{6.1}$$

From Table 6.3 we can observe that the p-values in five hypotheses tests

were very low ($p < \alpha = 0.0083$); in these tests we rejected the null hypothesis. Thus, in these hypotheses tests, we can assume that the difference between the MAP@5 for the voting ensemble and individual algorithms are statistically significant. In addition, we compared the ranking ensemble with the voting ensemble, but for this statistical test we obtained a p-value of 0.23, so we did not reject the null hypothesis. Therefore, we cannot assume that there is a statistically significant difference between the MAP@5 of the voting and ranking ensemble.

Table 6.3: The p-values and conclusion of the statistical tests

algorithms that were tested	p-value	conclusion
tf vs voting ensemble	4.455e-05	reject H_0
tf-idf vs voting ensemble	3.0000e-04	reject H_0
TextRank vs voting ensemble	1.0000e-04	reject H_0
PositionRank vs voting ensemble	3.0000e-04	reject H_0
MultipartiteRank voting ensemble	0.0000e+00	reject H_0
voting ensemble vs ranking ensemble	0.23	do not reject H_0

6.2 Results on the short and long documents

Recall that our fifth sub-question is to determine if the algorithms are robust to changes in the search space. More specifically:

SQ5: Does the length of an article have an effect on the performance of the unsupervised algorithms?

To answer this sub-question we provide the results for short and long documents. We only show the results for the MAP@5 and $F1$ score@5. Table 6.5 shows the results obtained from testing the algorithms on the short and long documents. From these results, we can observe that the voting ensemble obtained the best results based on the MAP@5. Interestingly, the long documents performed better in terms of the MAP@5, which suggests that the voting ensemble is robust for changes in the search space.

In short, we conclude that each algorithm performed differently to changes in the search space. For example, the ensemble methods, tf-idf and TextRank performed better, but many individual algorithms (e.g., tf), obtained a lower MAP@5.

Table 6.4: Results of precision@5 and MAP@5 on the combined *extractive* gold standard for short documents

models	short documents MAP@5%	long documents MAP@5%
tf	52.5 (41.8, 67.3)	45.1 (36.1, 56.4)
tf-idf	48.1 (39.9, 60.7)	53.3 (43.6, 67.4)
TextRank	43.0 (32.8, 55.7)	44.9 (35.4, 57.8)
PositionRank	50.6 (41.9, 62.6)	47.7 (38.1, 59.9)
MultipartiteRank	44.4 (34.1, 57.0)	41.9 (33.2, 52.9)
ranking ensemble	52.4 (43.8, 61.3)	54.3 (46.3, 63.5)
voting ensemble	**57.8 (47.8, 68.3)**	**58.7 (50.1, 67.8)**

Table 6.5: Results of $F1$ score on the individual gold standard documents of the *extractive* keyphrases

models	short documents F1-score@5%	long documents F1-score@5%
tf	27.0 (21.8, 33.4)	24.4 (19.2, 30.3)
tf-idf	23.9 (19.2, 29.5)	25.8 (20.6, 32.4)
TextRank	20.1 (15.3, 25.8)	24.0 (18.9, 29.8)
PositionRank	23.0 (17.0, 30.0)	26.0 (21.0, 32.4)
MultipartiteRank	24.6 (19.7, 30.9)	23.9 (19.3, 29.2)
ranking ensemble	26.2 (21.4, 30.7)	**28.8 (23.6, 33.8)**
voting ensemble	**28.9 (23.9, 34.0)**	28.7 (24.1, 33.5)

6.3 Analysis

6.3.1 Anecdotical evidence

SQ6: Can the use of a domain-specific thesaurus supplement the best-performing unsupervised algorithm, by extracting domain-specific abstractive keyphrases?

Evaluating abstractive keyphrases by using a gold standard is challenging as was discussed in Section 5.3.4. For this reason, we decided to not evaluate the *abstractive* keyphrases by using automated evaluation metrics. Instead, we provide anecdotical evidence that graph databases can enrich our current set of keyphrases. Below we provide a sample article, which belonged to the category of short documents. The phrases that are boldfaced are keyphrases according to the voting ensemble, and the underlined phrases are keyphrases according to the combined gold standard. For this specific document, the annotators did not

annotate any abstractive keyphrases.

Recall that in Section 5.5 we mentioned that we would define X after observing the experimental results. After observing the results, we let X to be all the keyphrases of the individual algorithms (obtaining 30 keyphrases for each document).

Hitteprotocol toepassen in je werk: hoe doe je dat?

Bij warm weer hebben vooral ouderen en mensen die hulpbehoevend zijn kans op ademhalingsproblemen, jeuk, duizeligheid en levensbedreigende aandoeningen. Daarom komt het Rijksinstituut en Milieu RIVM met het **Nationaal Hitteplan**. Hierin staan ook maatregelen voor zorginstellingen om oververhitting en uitdroging te voorkomen. Hoe ondersteun je kwetsbare mensen tegen de **warmte**, zowel overdag als in de nacht? Lees onderstaande **tips**:

1. Drink voldoende;
2. Draag dunne kleding die bescherming biedt tegen verbranding door de zon;
3. Zoek de schaduw op;
4. Smeer de huid in met zonnebrandcrème;
5. Beperk lichamelijke inspanning in de middag;
6. Houd de woning koel door tijdig zonwering, ventilator of airconditioning te gebruiken;
7. Let extra op **mensen** in je omgeving die zorg nodig hebben.

Hitteprotocol

- Hoeveel zonnebrand moet je op smeren en hoe vaak?
- Hoe stel je een zonnesteek vast?
- Wat doe je bij een zonnesteek?

We zien dat zorgorganisaties in de gehandicaptensector op basis van het **Nationaal Hitteplan** vaak een **hitteprotocol** maken voor in de eigen organisatie. Bekijk het **hitteprotocol** van zorgorganisatie Bartiméus.

Source: www.kennispleingehandicaptensector.nl

For this specific article we extracted the top five keyphrases from our voting ensemble, and combined this with the approach of extracting domain-specific abstractive keyphrases. For this specific example the phrases "kwetsbare burgers", and "ouderen" both appeared in X, and "kwetsbare ouderen" is a broader term of both "kwetsbare burgers", and "ouderen'. Therefore, it was possible

to extract the abstractive keyphrase "kwetsbare ouderen". Although we could not evaluate this phrase with the current evaluation approach it is worth mentioning that "kwetsbare ouderen" is semantically very similar to an extractive keyphrases in the gold standard: *ouderen en mensen die hulpbehoevend zijn*. Thus, these results suggest that our current approach of extracting abstractive keyphrases could enrich the keyphrase set of the algorithm.

Furthermore, there is only one true positive in this article: hitteprotocol. However, this example also illustrates the limitations of the exact match restriction. Namely, one keyphrase of our algorithm was similar to the gold standard, but it was still classified as a false positive. Namely, the gold standard contained the keyphrase "hitteplan", but our algorithm extracted "nationaal hitteplan"; a keyphrase that is arguably more accurate.

To conclude, we extracted 500 keyphrases with our algorithms on the final test set. It should be stressed that all keyphrases were evaluated on the original keyphrases in the text (i.e., including non-descriptors and no broader terms). However, incorporating a graph database with a domain-specific thesaurus allows us to change 95 of these extractive keyphrases from their non-descriptors to descriptors. Furthermore, eight keyphrases could be added because the broader term appeared twice in the final set of X keyphrases. Thus, this approach has the potential to greatly enrich current unsupervised algorithms.

6.3.2 Error-analysis

- **Over-extraction errors**: The majority of the errors of the algorithms are over-extraction errors, which means that the algorithms extracts terms that are semantically similar. For example, on a certain document it extracted "hoge eisen" as the second keyphrase, and "eisen" as the third keyphrase. Obviously, these terms are very similar, and we argue that keyphrase "eisen" should not have been extracted due to it being semantically very similar to "hoge eisen". An algorithm such as MultipartiteRank did not suffer from this shortcoming because it focuses on topical diversity (as discussed in 4.2.5). Nonetheless, the results indicate that this does not necessarily improve the accuracy of the model.

- **Person names**: Many documents in our test set often referred to specific persons within long-term care. The ground truth frequently contained the full name of this person to denote its importance. Even though our systems often extracted these persons as well, in many cases they did not extract the full name, but only their first- or family name. This is caused by the fact that many documents first reference to the full name, but thereafter only mention their first or last name. For this reason, the first or last name frequently appears in the document, but their full name does not.

- **Inability of extracting longer keyphrases**: Many keyphrases extractors are only capable of extracting keyphrases unigrams, bigrams, and trigrams. As a result, all keyphrases in the gold standard that are longer

than three words are classified as a false negative. For example, in Section 6.3.1 one keyphrase according to the gold standard was: *ouderen en mensen die hulpbehoevend zijn*, which our voting ensemble was unable to extract.

- **Infrequency errors**: Many unsupervised algorithms rely on using the frequency of a word to determine its importance. Therefore, it is still relatively difficult for unsupervised algorithms to find relevant terms that only appear once or twice in the document. This can be clearly illustrated in the example in Section 6.3.1 where many of the keyphrases according to the gold standard are not selected because they only appear once (e.g., oververhitting). There are some algorithms such as TextRank that put less emphasis on the frequency of a word to determine its final score. However, Section 6.1 leads us to conclude the approach of this algorithm does not improve the performance. In general, this leads us to conclude our algorithms perform relatively well on documents where keyphrases appear frequently, but less well on documents where keyphrases appear infrequently.

- **Evaluation errors**: There are certain false positives, which are not necessarily errors of the algorithms but can be contributed to restrictions of the exact match. In fact, for many keyphrases, we found our algorithms were more detailed than the gold standard but were still penalized. An example of this was given in Section 6.3.1 where our algorithm extracted "nationaal hitteplan", but the gold standard only contained "hitteplan".

Chapter 7

Conclusion and discussion

This thesis had one research objective, in this section we answer if the research goal was accomplished. Furthermore, we summarize our work and determine the significance of this research for the field. Additionally, in Section 7.1 we provide two limitations of this research, and in Section 7.2 we suggest topics for future research.

Recall that the main objective of this thesis was as follows:

Developing an unsupervised algorithm that can accurately extract keyphrases, for documents in the long-term care sector.

In summary, we can conclude that this research goal was achieved. Namely, the state-of-the-art unsupervised methods are implemented, and the voting ensemble even achieved better results than the current state-of-the-art. Nonetheless, we do need to mention one caveat, which is that the voting ensemble achieved a F_1 score of 28.8% on the extractive keyphrases, but the average mutual F_1 score among the annotators was higher with a F_1 score of 35.1%. Accordingly, these results indicate there is still room for improvement.

Nevertheless, we argue that it is still possible to utilize the proposed keyphrase extraction methods for Vilans. Namely, our voting ensemble performs better than tf-idf, which is currently used for their information systems. Also, current methods are unable to extract relevant abstractive keyphrases, but our results suggest that our methods may solve this difficult problem. Therefore, we strongly believe our methods are more appropriate than current methods that are used for indexing documents. As a result, this could empower faster and more informed decision making for healthcare professionals.

However, it should be noted that without the inference of employees our approaches are still not suitable for summarization purposes on their website. Namely, there are still some errors in the system, which Section 6.3.2 focused on. Therefore, for summarization purposes we recommend Vilans employees to quickly validate the keyphrases of the algorithm after writing a document.

The main implications of this research

To sum up, our work has introduced four major findings. Firstly, we hypothesized that developing an unsupervised ensemble method could improve current state-of-the-art unsupervised algorithms. Our results appear promising because our voting ensemble provided a relative improvement of 14.7% in terms of MAP@5 with the current state-of-the-art solutions; these results were also statistically significant. Furthermore, the voting ensemble appeared to be quite robust for variations in the search space because it outperformed individual methods on short and long documents. These results appear promising because two problems in this field are developing accurate unsupervised algorithms, and handling longer documents Hasan and Ng (2014).

Secondly, we found a solution for extracting abstractive keyphrases by combining a domain-specific thesaurus with graph databases. To our knowledge, this is the only approach so far which is able to extract potentially relevant abstractive keyphrases in an unsupervised setting, The work by Yuan et al. (2018) and Meng et al. (2017) focused on extracting abstractive keyphrases, but we strongly believe our method is better suited for practical applications because it does not rely on a large annotated corpus. Our method does rely on a domain-specific thesaurus, but within many domains this is available. The results are difficult to quantify because evaluating abstractive keyphrases with a gold standard is difficult, as explained in Section 5.3.4. Nevertheless, anecdotical evidence suggests that the approach is promising.

Thirdly, we have focused on implementing different approaches for calculating agreement measures in keyphrases extraction. In short, the results of these agreement measures reinforce the claims by Medelyan and Witten (2006); Nguyen and Kan (2007); Sterckx et al. (2018): keyphrase extraction is a subjective task. As mentioned in 5.3.4, we argue that the subjectivity of keyphrases extraction could be a major reason for it performing worse than many other NLP tasks. Therefore, we propose future researchers to keep this in mind when framing their research question, or research goal. For instance, instead of developing an algorithm that can "accurately" extract keyphrases it is more interesting to investigate how the algorithms perform in comparison to human performance. Alternatively, one could also utilize a different experimental setup. Namely, we decided to compare multiple algorithms, so we decided to determine a golden standard by letting domain-experts annotate keyphrases. However, it is also possible to limit the number of algorithms in a study and utilize human evaluation, which is further discussed in Section 7.2.

Lastly, our experimental results showed that many state-of-the-art unsupervised algorithms provide rather unsatisfactory performance. We were surprised that graph-based algorithms obtained worse results than the statistical algorithms on the combined set of documents. To our knowledge, there is no research that used tf as a baseline, so it is hard to compare these results with other research. Nevertheless, the results indicate that tf can be used as an effective baseline, if there is not a big domain-specific corpus available to determine the weight of each word.

Moreover, our experiments confirm the findings of Sterckx et al. (2018) and Hasan and Ng (2010): tf-idf can outperform graph-based algorithms. However, it contradicts the findings of Yuan et al. (2018) and Florescu and Caragea (2017), were graph-based algorithms outperformed tf-idf.

To conclude, the experimental results of Meng et al. (2017); Sterckx et al. (2018); Yuan et al. (2018) showed the superior effectiveness of supervised models over unsupervised algorithms, which exposes a major weakness of this field: there is lack of accurate unsupervised algorithms. In many practical applications applying supervised learning in infeasible because acquiring labeled data is expensive and time-consuming. Thus, we hope that this research can be helpful in extracting accurate keyphrases when labeled data is unavailable. Furthermore, our ensemble methods are unsupervised, so we suspect similar performance on data sets from different domains and languages. Nonetheless, we encourage future researchers to validate these suspicions.

7.1 Limitations of this research

It is plausible that two limitations may have influenced the results obtained. The first is regarding errors during the annotation process, and the second is regarding our small sample size. Both limitations highlight the difficulty of creating a gold standard in keyphrase extraction.

Annotation process

As explained in Section 5.2 we were required to make several adjustments to the gold standard, but we cannot rule out that these changes might have slightly altered the results. More specifically, we believe that the 2.91% typographical errors had no influence on the results because these were manually changed and kept in the same category. However, we cannot claim the same regarding the 3.72% data-inconsistencies, 0.36% annotated keyphrases that were deleted, and 3.86% keyphrases that were moved to the abstractive column.

For instance, many keyphrases with regards to the data inconsistencies were phrases that were contained in the subtitle. Therefore, because these phrases were early in the document we do believe the performance of approaches that favor phrases at the beginning of the document (e.g., PositionRank) would slightly improve.

Future researchers can prevent these problems by periodically validating the gold standard during its creation. This would allow a researcher to spot miscommunications early in the annotation process. Alternatively, human evaluation could be utilized; we suspect that a human evaluation task description is more straightforward, which may lead to fewer errors during the annotation process.

Sample size

The small validation and test set made it hard to compare the performance in detail. More specifically, we believe our ranking ensemble would perform better if we had a larger sample size for our validation set. Also, comparing the performance on short versus long documents was difficult because we were restricted to a sample size of 50 documents for short and long documents.

We made use of this small evaluation set to limit the effort and time required by the employees of Vilans. Nevertheless, further research could try to make a more solid comparison by increasing the number of documents annotated by employees with domain knowledge. Furthermore, we believe increasing the number of employees creates a more reliable estimation of the keyphrases.

7.2 Future research topics

Combining a domain-specific thesaurus with a lexical database

Hasan and Ng (2014) recommended the utilization of background knowledge such that keyphrase extractors could have a deeper "understanding" of a document. Our approach incorporated these recommendations through utilizing a domain-specific thesaurus, and although our results appear promising we believe there is much more potential. Namely, our approach was limited to utilizing a domain-specific thesaurus, but we suspect our approach could be extended to incorporate external lexical databases. For example, BabelNet (Navigli and Ponzetto, 2012) is a lexical database, which provides many semantic relations in 271 languages including Dutch. We strongly believe combining this external database with graph databases could be used in future research to capture intrinsic semantic relationships in the human language. As a result, the final set of current keyphrase extractors could be greatly improved.

Exploring alternative evaluation methods

After conducting this research we strongly believe many studies would benefit from using human evaluation instead of creating a gold standard. Namely, we believe there are three main problems with using a gold standard.

Firstly, it is infeasible to compare abstractive keyphrases by using a gold standard, Namely, Section 5.3.4 showed that annotators have little to no agreement by looking at the *abstractive* keyphrases they had in common. In contrast, there was much more similarity by looking at the *executive* keyphrases. This is unsurprising because there is much variability in the human language. For this reason, we initially wanted to evaluate our algorithms only on extractive keyphrases, but after conducting a pilot version the annotators also wanted to write down keyphrases that are not present in the document. Nevertheless, after performing this study we believe evaluating keyphrases on asbtractive keyphrases can lead to many unjustified false negatives.

Secondly, as illustrated in Section 6.3.1 the exact match is very strict, which often leads to unjustified false positives.

Thirdly, many current evaluation setups the algorithms are penalized if the number of extracted keyphrases is not the same as the gold standard. To illustrate this problem, We denote A by the actual set of keyphrases according to the algorithm and G is the actual set of keyphrases according to the gold standard. Furthermore, let $|A|$ be the cardinality (the number of elements) of set A, and let B be the cardinality according to the golden annotations. Thus, if $|A| \leq |G|$ a recall of 1 is unattainable, even if $A \cap B = 1$. This is due to the fact that every keyphrase in the set $(A \cap B)c$ (the complement of $A \cap B$), is classified as a false negative. In contrast, if $|A| \geq |G|$ then a precision of 1 is unattainable, even if $A \cap B = 1$ (i.e., the keyphrases of the algorithm and golden annotations are exactly the same). This is due to the fact every keyphrase in the set $(A \cap B)c$, is classified as a false positive. Thus, the only scenario where recall and precision could theoretically achieve a recall and precision of 1 is when $|A| \leq |G|$ and $|G| \leq |A|$, such that $|A| = |G|$. Evaluating the algorithms on not extracting the right number of keyphrases is questionable because Section 5.3.1 illustrated that the number of keyphrases per document also varies much per annotator. Therefore, it is debatable if there is actually such a thing as an "optimal" number of keyphrases for a document.

Yuan et al. (2018) also notified this problem. Furthermore, they proposed a new evaluation metric: $F_1@|G|$, such that $|A| = |G|$ for all documents in the test set. Thus, in theory, this allows the keyphrase extraction algorithm to obtain an F_1 score equal to 1, if and only if $A \cap B = 1$. It should be noted that this elegantly solves the problem, but does assume knowledge of the gold standard that an algorithm would not have in a real-world scenario.

In this research the third problem described could be prevented; however, the third problem became apparent in the second evaluation approach, where we had to calculate the recall of the systems.

Therefore, we encourage future researchers to determine the precision of keyphrase extraction algorithms by human evaluation, if the resources allow doing so. Human evaluation could be implemented by using a binary for denoting if a phrase of an algorithm is actually a keyphrase. Thereafter, metrics in this research such as precision@5, or MAP@5 could be used.

Combining our ensemble methods with semi-supervised learning

Ye and Wang (2018) proposed an approach of utilizing semi-supervised learning for keyphrase extraction. Their results appeared promising because they achieved state-of-the-art results, but required much less labeled data than the DNNs discussed in Section 3.2.4. However, their methods relied on synthetic keyphrases obtained from unsupervised algorithms. Therefore, we argue that our voting ensemble can also be used for creating accurate synthetic keyphrases for semi-supervised learning.

Bibliography

Vilans over ons. https://www.vilans.nl/over-ons, 2019. Accessed: 2020-05-11.

Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555. Association for Computational Linguistics, August 2017.

Marco Basaldella, Elisa Antolli, Giuseppe Serra, and Carlo Tasso. Bidirectional lstm recurrent neural network for keyphrase extraction. In *Italian Research Conference on Digital Libraries*, pages 180–187. Springer, 2018.

Gábor Berend and Richárd Farkas. Sztergak: Feature engineering for keyphrase extraction. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 186–189. Association for Computational Linguistics, 2010.

Carlo Bonferroni. Teoria statistica delle classi e calcolo delle probabilita. *Pubblicazioni del R Istituto Superiore di Scienze Economiche e Commericiali di Firenze*, pages 3–62, 1936.

Florian Boudin. Pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 69–73, 2016.

Florian Boudin. Unsupervised keyphrase extraction with multipartite graphs. *arXiv preprint arXiv:1803.08721*, 2018.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. Topicrank: Graph-based topic ranking for keyphrase extraction. In *International Joint Conference on Natural Language Processing (IJCNLP)*, pages 543–551, 2013.

Paul R Cohen. *Empirical methods for artificial intelligence*, volume 139. MIT press Cambridge, MA, 1995.

Anthony Christopher Davison and David Victor Hinkley. *Bootstrap methods and their application*. Number 1. Cambridge university press, 1997.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392, 2018.

Jacob Eisenstein. *Introduction to Natural Language Processing*. MIT press, 2018.

Halil Ibrahim Erdal and Onur Karakurt. Advancing monthly streamflow prediction accuracy of cart models using ensemble learning paradigms. *Journal of Hydrology*, 477:119–128, 2013.

Corina Florescu and Cornelia Caragea. Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1115, 2017.

David F Gleich. Pagerank beyond the web. *SIAM Review*, 57(3):321–363, 2015.

Kazi Saidul Hasan and Vincent Ng. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 365–373. Association for Computational Linguistics, 2010.

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)*, pages 1262–1273, 2014.

Wassily Hoeffding. The large-sample power of tests based on permutations of observations. *The Annals of Mathematical Statistics*, pages 169–192, 1952.

Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223. Association for Computational Linguistics, 2003.

Anette Hulth and Beáta B Megyesi. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 537–544. Association for Computational Linguistics, 2006.

Xin Jiang, Yunhua Hu, and Hang Li. A ranking approach to keyphrase extraction. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 756–757, 2009.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. Automatic keyphrase extraction from scientific articles. *Language resources and evaluation*, 47(3):723–742, 2013.

Florian Kunneman, Christine Liebrecht, Margot Van Mulken, and Antal Van den Bosch. Signaling sarcasm: From hyperbole to hashtag. *Information Processing & Management*, 51(4):500–509, 2015.

Daniel T Larose and Chantal D Larose. *Discovering knowledge in data: an introduction to data mining*, volume 4. John Wiley & Sons, 2014.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 366–376. Association for Computational Linguistics, 2010.

Edward Loper and Steven Bird. Natural language processing with python. 2009.

Christopher D Manning. Part-of-speech tagging from 97% to 100%: is it time for some linguistics? In *International conference on intelligent text processing and computational linguistics*, pages 171–189. Springer, 2011.

Yann Mathet, Antoine Widlöcher, and Jean-Philippe Métivier. The unified and holistic method gamma (γ) for inter-annotator agreement measure and alignment. *Computational Linguistics*, 41(3):437–479, 2015.

Olena Medelyan and Ian H Witten. Thesaurus based automatic keyphrase indexing. In *Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pages 296–297, 2006.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*, 2017.

Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411, 2004.

Ron C Mittelhammer, George G Judge, and Douglas J Miller. *Econometric foundations pack with CD-ROM*. Cambridge University Press, 2000.

Roberto Navigli and Simone Paolo Ponzetto. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250, 2012.

Ani Nenkova and Lucy Vanderwende. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005*, 101, 2005.

Thuy Dung Nguyen and Min-Yen Kan. Keyphrase extraction in scientific publications. In *International conference on Asian digital libraries*, pages 317–326. Springer, 2007.

Eric W Noreen. *Computer-intensive methods for testing hypotheses*. Wiley New York, 1989.

David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12: 2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D Manning. Stanza: A python natural language processing toolkit for many human languages. *arXiv preprint arXiv:2003.07082*, 2020.

Mark D Smucker, James Allan, and Ben Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 623–632, 2007.

Lucas Sterckx, Thomas Demeester, Johannes Deleu, and Chris Develder. Topical word importance for fast keyphrase extraction. In *Proceedings of the 24th International Conference on World Wide Web*, pages 121–122, 2015.

Lucas Sterckx, Thomas Demeester, Johannes Deleu, and Chris Develder. Creation and evaluation of large keyphrase extraction collections with multiple opinions. *Language Resources and Evaluation*, 52(2):503–532, 2018.

Peter D Turney. Learning algorithms for keyphrase extraction. *Information retrieval*, 2(4):303–336, 2000.

Julián Urbano, Harlley Lima, and Alan Hanjalic. Statistical significance testing in information retrieval: an empirical analysis of type i, type ii and type iii errors. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 505–514, 2019.

Xiaojun Wan. Using bilingual knowledge and ensemble techniques for unsupervised chinese sentiment analysis. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 553–561, 2008.

Xiaojun Wan and Jianguo Xiao. Collabrank: towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 969–976, 2008.

Hai Ye and Lu Wang. Semi-supervised learning for neural keyphrase generation. *arXiv preprint arXiv:1808.06773*, 2018.

Wen-tau Yih, Joshua Goodman, and Vitor R Carvalho. Finding advertising keywords on web pages. In *Proceedings of the 15th international conference on World Wide Web*, pages 213–222, 2006.

Xingdi Yuan, Tong Wang, Rui Meng, Khushboo Thaker, Peter Brusilovsky, Daqing He, and Adam Trischler. One size does not fit all: Generating and evaluating variable number of keyphrases. *arXiv preprint arXiv:1810.05241*, 2018.

Qi Zhang, Yang Wang, Yeyun Gong, and Xuan-Jing Huang. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 836–845, 2016.

Yongzheng Zhang, Evangelos Milios, and Nur Zincir-Heywood. A comparative study on key phrase extraction methods in automatic web site summarization. *J. Digit. Inf. Manag.*, 5(5):323–332, 2007.

Appendix A

Vilans: connecting the scientific and healthcare community

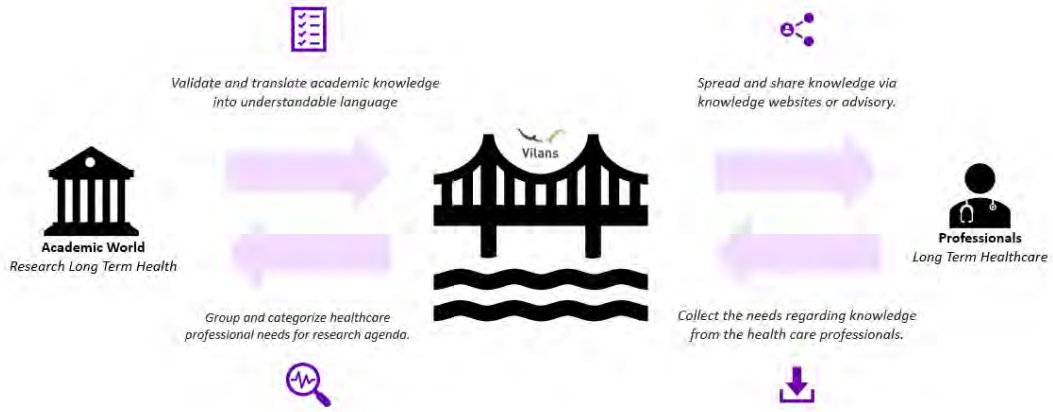


Figure A.1: An illustration of how Vilans connects the scientific community with health care professionals.

Appendix B

The data preprocessing

At first glance it appears the data is unstructured because the data is obtained from the different Web sources. More specifically, there are two main problems with the data before we can implement the algorithms described in Section 4. The first is that some articles are encoded in Windows-1252, while others use the more common UTF-8 standard. [1] This is problematic because this means not all characters are represented in the same format. To ensure all documents use the UTF-8 we need to translate the Windows-1252 characters into Unicode, a process called decoding. After this the characters need to be translated from Unicode into UTF-8, a process called encoding. This process is illustrated in Figure B.1. For the encoding and decoding we used Pandas, which is an external open-source library in Python.

In addition, all the documents were also in HTML format. The HTML tags were deleted with BeautifulSoup and afterwards regular expressions were used to delete irrelevant characters (e.g., line breaks). The appendix gives three examples of how the text files looked before and after this process

Lastly, across all files the actual content of the articles is in different columns. For each individual file it was determined in which column the content of the article was contained, and the rest of the columns are deleted. Furthermore, there are a total of 19 duplicate articles across the files, these are dropped.

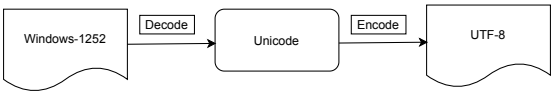


Figure B.1: The process of converting the windows-1252 characters to UTF-8 through Unicode.

[1] For more details about Windows-1252, Unicode and UTF-8 see: https://www.nltk.org/book/ch03.html

Appendix C

Vilans data

Although all these sources focus on the long-term care each data source discusses different topics. For an elaborate analysis of the data source and their differences we refer to the official webpage of Vilans.[1]

Table C.1 illustrates the number of articles for each year in the dataset. From this table we can see that most articles are published between 2014 and 2019, but there are also articles from 2013 and 2020. There was also an article from 1980, but this omitted from the graph since it is assumed there was an error when registering this date.

Table C.1: The data sources for this research and their corresponding number of articles

data source	mean
agenda items gehandicaptensector	189
nieuws items gehandicaptensector	4 122
dementiezorg voor elkaar	359
lennisinfrastructuur langdurige zorg	95
vilans.nl	1 025
waardigheidentrots	2 888
Corpus	8 678

In Section 3.7 we explained that one research sub-question will determine if the algorithms perform better for shorter or longer articles. Therefore, we would also need to know the length distribution of the data sources, so we have an indication of what constitutes a short and long article. In Figure C.2 the length distribution of the whole corpus, and on an individual level is illustrated. From the figure on the left it becomes clear that the length distribution is right-skewed; i.e., most articles are relatively short, but there are also outliers (>1000

[1] For more information visit: https://www.vilans.nl/websites

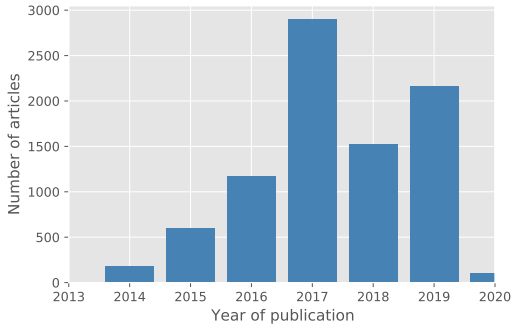
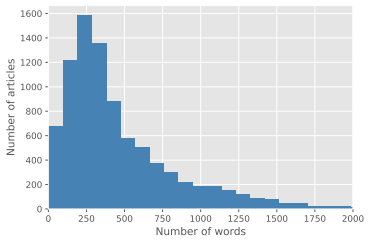


Figure C.1: Distribution of the articles over the years

words) in the data set. From the figure on the right it becomes clear that each data source contains outliers. Especially the articles from Vilans.nl and waardigheidentrots are relatively long. It should be noted that extreme outliers (>1000 words) are omitted from these graphs for clarity purposes.

Figure C.2: The length distribution of the whole corpus and on an individual level.

(a) Length distribution of the whole corpus



(b) The number of words for each individual data source

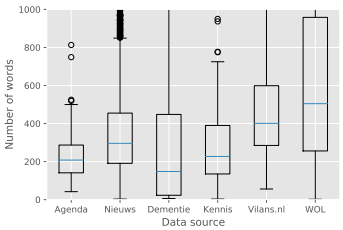


Table C.2: The mean, median, minimum, maximum and standard deviation for each data source and the whole corpus.

Data source	Mean	Median	Min.	Max.	STDEV.
Agenda items gehandicaptensector	227	208	42	1 334	138
Nieuws items gehandicaptensector	366	296	3	4 130	280
Dementiezorg voor elkaar	325	148	6	2 482	433
Kennisinfrastructuur langdurige zorg	322	227	3	1 921	306
Vilans.nl	477	401	56	3 748	310
Waardigheidentrots	642	504	2	22 727	631
Corpus	466	341	2	22 727	456

Appendix D

Changes to the gold standard

In brief, the errors of the original gold standard were as follows:

1. The annotations were not consistently ordered in relevance.
2. The annotations contained typos
3. Keyphrases that were assigned to the category "keyphrases in the text", often contained keyphrases outside the text.
4. For two articles the keyphrases did not correspond to any phrases in the original text. Our assumption is that the annotators accidently clicked on the wrong link.
5. Keyphrases often did not contain stop words.
6. Denoting semenatically equivalent keyphrases (leefstijl instead of levensstijl). These semantically similar keyphrases could also not resolved by using a stemmer.

Furthermore, there were some minor data-inconsistenties between the article on the website URL, and the data set. Therefore, all these keyphrases were assigned to the category "keyphrases outside the text"

The changes will are noted in the following format: index in Excel; keyphrases that was changed; reason for changing the keyphrase. For example, if we changed the following keyphrase "respijtzorgmogelijkheden" in line 3 of the Excel because the keyphrase was not in the original text, the changes will be registered as follows:

3; **respijtzorgmogelijkheden**; moved to column *keyphrases outside of the text* because the keyphrase was not in the original text.

Changes to first annotator

0; **participatief onderzoek**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

1; **EMB**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

1; **waargebeurde verhaa**; typo, which was changed to *waargebeurde verhaal*.

3; **respijtzorgmogelijkheden**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

10; **eigen invloed**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

10; **kinderwens ouderschap vg**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

20; **juridische kennis**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

20; **wie beslis**; typo, which was changed to *wie beslist*.

22; **probleemgedrag**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

23; **kinderwens en ouderschap**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

25; **juridische kennis**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

27; **EMB**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

28; **combinatie ziektebeelden**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

28; **combinatie ziektebeelden**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

30; **ziekenhuis**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

33; **relatie**; typo, which was changed to *relaties* because the annotated keyphrase was not in the original text.

39; **leefstijl**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

42; **verandering**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

44; **thuiswonend**; typo, which was changed to *thuiswonende* because the annotated keyphrase was not in the original text.

42; **kostenloos**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

44; **e learnings**; typo, which was changed to *e learning* because the annotated keyphrase was not in the original text.

52; **zorg en welzijn**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

54; **duur**; typo, which was changed to *duurste* because the annotated keyphrase was not in the original text.

60; **jongreren**; typo, which was changed to *jongeren* because the annotated keyphrase was not in the original text.

70; **Karlijk Kwint**; typo, which was changed to *Karlijn Kwint* because the annotated keyphrase was not in the original text.

77; **nee tenzj**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

89; **wezijn**; typo, which was changed to *welzijn* because the annotated keyphrase was not in the original text.

89; **pltaform**; typo, which was changed to *platform* because the annotated keyphrase was not in the original text.

94; **project wonen met gemak**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

94; **mogenlijkheden**; typo, which was changed to *mogelijkheden* because the annotated keyphrase was not in the original text.

97; **leidraad oud gelukkig**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

99; **emb**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

Changes to second annotator

1; **ernstige meervoudige beperking**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

11; **tweede kamer**; typo, which was changed to *eerste kamer* because the annotated keyphrase was not in the original text.

14; **organisaties voor mantelzorgondersteuning**; typo, which was changed to *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

14; **respijtwijzer.nl**; typo, which was changed to *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

15; **wo**; synonym mistake, which was changed to *universiteit* because the annotated keyphrase was not in the original text.

16; **professionals**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

17; **onderzoek**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

22; **ouders van een kind met een verstandelijke beperking**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

23; **ikbenhetkennisplein**; typo, which was changed to *ik ben het kennisplein* because the annotated keyphrase was not in the original text.

24; **website vilans**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

24; **cookies**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

26; **eerste kamer**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

42; **sluiting**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

42; **te vaak voorgeschreven**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

43; **richtlijn constructie afname zelfrapportagelijsten**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

44; **professionals**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

47; **toekomstige zorgprofessionals**; typo, which was changed to *toekomstige professionals* because the annotated keyphrase was not in the original text.

49; **coalitie Van begin tot het einde**; typo, which was changed to *coalitie Van betekenis tot het einde* because the annotated keyphrase was not in the original text.

50; **optimaal leefritme**; typo, which was changed to *optimaal persoonlijk leefritme* because the annotated keyphrase was not in the original text.

50; **complexe zorgvragen**; typo, which was changed to *complexere zorgvragen* because the annotated keyphrase was not in the original text.

51; **complexe zorgvragen**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

52; **certificaat**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

52; **free learning**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

52; **accreditatiepunten**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

53; **zorgvernieuwers 2018**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

53; **kennis uitwisselen**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

54; **ouderen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

56; **indiecatieloos**; typo, which was changed to *indicatieloze* because the annotated keyphrase was not in the original text.

66; **accountmanagers**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

68; **context interventies**; data inconsistency, which was moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the dataset.

69; **ervaringsverhaal**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

71; **hoe georganiseerd**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

73; **samenwerking**; typo, which was changed to *samenwerken* because the annotated keyphrase was not in the original text.

74; **67 hoogleraren en zorgprofessionals**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

79; **zes tips**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

80; **Active Assisted Living project**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

82; **geestelijke verzorger**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

82; **samenwerking**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

83; **veilig thuis wonen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

84; **programma (ont)regel de langdurige zorg**; typo, which was changed to *programma (ont)regel de zorg* because the annotated keyphrase was not in the original text.

84; **coronaondersteuning**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

85; **extra middelen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

85; **ervaringen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

86; **definitie calimiteit**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

88; **complexe zorgvragen**; typo, which was changed to *complexere zorgvragen* because the annotated keyphrase was not in the original text.

89; **ruimtelijke analyse van de wijk**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

89; **stichting eten+welzijn**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

89; **oprichten**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

92; **gesprek aangaan**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

95; **niet van bovenaf opleggen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

96; **aan het werk in het verpleeghuis**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

99; **Europees onderzoeksproject SUSTAIN**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

99; **24 uurs ritme**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

99; **optimaal leefritme**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

Changes to third annotator

0; **participatief onderzoek**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

3; **mantelzorg**; typo, which was changed to *mantelzorgers* because the annotated keyphrase was not in the original text.

3; **mantelzorgondersteuning**; typo, which was changed to *ondersteuning mantelzorg* because the annotated keyphrase was not in the original text.

3; **vervanging**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

6; **verpleeghuis**; typo, which was changed to *verpleeghuiszorg* because the annotated keyphrase was not in the original text.

9; **kinderwens**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

12; **open dag**; typo, which was changed to *open dagen* because the annotated keyphrase was not in the original text.

16; **migranten**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

17; **subsidie**; typo, which was changed to *stimuleringssubsidie* because the annotated keyphrase was not in the original text.

18; **verpleeghuis**; typo, which was changed to *verpleeghuissector* because the annotated keyphrase was not in the original text.

18; **hygiene**; typo, which was changed to *hygienemaatregelen* because the annotated keyphrase was not in the original text.

19; **antibioticaresistentie, verpleeghuis, hygiene, hygienemaatregelen**; **deleted** because none of the annotated keyphrase were in the original text we assumed the annotator accidentally clicked on the wrong URL.

20; **samen beslissen**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

21; **juridisch**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

23; **medisch**; typo, which was changed to *medische* because the annotated keyphrase was not in the original text.

26; **21 miljard**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

27; **hechtingsproblemen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

27; **sociale problemen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

29; **hygiene**; typo, which was changed to *hygienecode* because the annotated keyphrase was not in the original text.

32; **Niet-Aangeboren-hersenletsel (NAH)**; typo, which was changed to *NAH* because annotators were encouraged to write down abbreviations, if these and not the full words.

32; **Niet-Aangeboren-hersenletsel (NAH)**; typo, which was changed to *NAH* because annotators were encouraged to write down abbreviations, if these and not the full words.

32; **Ernstige Meervoudige Beperkingen (EMB)**; typo, which was changed to *EMB* because annotators were encouraged to write down abbreviations, if these and not the full words.

32; **Licht Verstandelijke Beperking (LVB)**; typo, which was changed to *LVB* because annotators were encouraged to write down abbreviations, if these and not the full words.

34; **ouderschap**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

34; **verstandelijke beperking**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

35; **gezonde leefstijl**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

36; **integrale zorg**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

42; **informatie**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

46; **medisch kindzorg systeem**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

51; **complexe zorgvraag**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

52; **elearning**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

60; **effectief**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

61; **indicatieloos**; typo, which was changed to *indicatieloze* because the annotated keyphrase was not in the original text.

63; **patronen blootleggen**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

68; **kwaliteitskader gehandicaptenzorg**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

68; **kennisontwikkeling**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

70; **praktijkkennis**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

71; **kwaliteit verbeteren**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

73; **erstandelijke beperking**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

74; **ondersteuning**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

77; **nee tenzij**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

78; **ontwikkeling**; typo, which was changed to *ontwikkelingen* because the annotated keyphrase was not in the original text.

80; **elearning**; data inconsistency *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

84; **corona**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

92; **kwaliteitskader vepleeghuiszorg**; typo, which was changed to *kwaliteitskader verpleeghuiszorg* because the annotated keyphrase was not in the original text.

92; **praktijkonderzoek**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

95; **gezondheid medewerkers**; typo, which was changed to *gezondheid zorgmedewerkers* because the annotated keyphrase was not in the original text.

95; **onderzoek**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.

99; **Mijn leven mijn ritme**; deleted because the annotated keyphrase was not in the original text, and **Mijnlevenmijnritme** was already annotated.

99; **dagstructuur**; moved to column *keyphrases outside of the text* because the annotated keyphrase was not in the original text.