Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction

Article · May 2004

1 author:

Anette Hulth
Public Health Agency of Sweden
52 PUBLICATIONS **1,312** CITATIONS

Combining Machine Learning and Natural Language Processing for Automatic Keyword Extraction

Anette Hulth

Doctoral Dissertation
Department of Computer and Systems Sciences
Stockholm University
April, 2004

Stockholm University/Royal Institute of Technology
Report series
No. 04-002
ISBN 91-7265-894-0
ISSN 1101-8526
ISRN SU-KTH/DSV/R--04/2--SE

Typeset by the author using LaTeX
Printed by Akademitryck AB, Edsbruk, Sweden, 2004

Abstract

Automatic keyword extraction is the task of automatically selecting a small set of terms describing the content of a single document. That a keyword is extracted means that it is present verbatim in the document to which it is assigned. This dissertation discusses the development of an algorithm for automatic keyword extraction, and presents a number of experiments, in which the performance of the algorithm is incrementally improved.

The approach taken is that of supervised machine learning, that is, prediction models are constructed from documents with known keywords. Before any learning can take place, the data must be pre-processed and represented. In the presented research, two problems concerning the representation for keyword extraction are tackled. Since a keyword may consist of more than one token, the first problem concerns where a keyword begins and ends in a running text, that is, how a candidate term is defined. In this dissertation, three term selection approaches are defined and evaluated. The first approach extracts all uni-, bi-, and trigrams, the second approach extracts all noun phrase chunks, while the third approach extracts all terms matching any of a number of empirically defined part-of-speech patterns.

Since the majority of the extracted candidate terms are not keywords, the second problem concerns how these terms can be limited, to only keep those that are appropriate as keywords. In the presented research, four features for filtering the candidate terms are investigated. These are term frequency, inverse document frequency, relative position of the first occurrence, and the part-of-speech tag or tags assigned to the candidate term.

The research presented in this dissertation is linguistically oriented in the sense that the output from natural language processing tools is a considerable factor both for the pre-processing of the data, as well as for the performance of the prediction models. Of the three term selection approaches, the best individual performance — as measured by keywords previously assigned by professional indexers — is achieved by the noun phrase chunk approach. The part-of-speech tag feature dramatically improves the performance of the models, independently of which term selection approach is applied. The highest performance is, however, achieved when the predictions of all three models are combined.

Keywords

(These keywords have been automatically extracted from the title and the abstract, by the algorithm described in this dissertation.)

Natural language processing, Prediction models, Automatic keyword extraction, Combining machine learning, Extracted candidate terms, Term frequency, Presented research, First problem concerns, Noun phrase chunk approach, Single document

"Every year new societies and new journals
are started in various parts of the world,
so that it becomes daily more difficult for workers
to keep themselves au courant *with the work of others."*

Henry B. Wheately, Hon. sec. of the Index Society,
and treasurer of the Early English Text Society, 1878

Acknowledgements

First of all, I would like to thank my supervisor Lars Asker for taking me on as a PhD student and for always being encouraging. I am indebted to Jussi Karlgren for all the inspiration and constructive input he has given me; without it I would not have come this far. I have had many fruitful discussions with Henrik Boström and Harko Verhagen, and they have also provided invaluable support during the last two years, for that I am most grateful. Many thanks to Beáta Megyesi for being a very good friend (with all that this implies), and for commenting and discussing my work. Thanks also to Jakob Tholander without whom I would have given up a long time ago.

In 2002, I had the opportunity to spend six stimulating months at TNO TPD in Delft, the Netherlands. The work I did while there have directly and indirectly shaped the research discussed in this dissertation. I would especially like to thank Wessel Kraaij, who also invited me, and Martijn Spitters.

The structure of the dissertation has benefited from discussions with Carl Gustaf Jansson. Thanks also to Björn Gambäck for providing the initial latex style sheets; to Anna Ståhl for excellent help with the design of the front cover; to Tony Lindgren for putting up with all my questions; to Rickard Cöster, Tessy Cerratto Pargman, Fredrik Kilander, and Hercules Dalianis for comments; Nikolaj Lindberg for helping me to write my first lines in Perl; and to Sara Rydin, Charles Schafer, Johnny Bigert, Magnus Sahlgren, and Jonas Sjöbergh for making some of my conference trips such good fun. I would like to express my gratitude to Fredrik Rutz, and Martin Eineborg for showing that there is a life also after DSV; and to Matei Ciobanu Morogan, Fresia Pérez, and Christian Schulte for sharing one of my hobbies. Thanks to Christian also for comments and latex support. There are also some people that I wish to thank, but who's name I do not know, namely the anonymous reviewers of my papers on which this dissertation is partly based.

My interest in this research topic has its roots in the master thesis work that Anna Jonsson and I did at the Swedish Institute of Computer Science (SICS) a number of years ago, where automatic keyword indexing was one of the problems we tackled. I would like to thank Anna for laughing both

with and at me. During the time at SICS, I got to know many interesting people, some of which are still my friends. Especially I would like to thank Fredrik Olsson, Kristofer Franzén, Fredrik Espinoza, and Mark Tierney.

Thanks for the technical support from DMC, especially from Sven Olofsson, and Niklas Brunbäck. I am grateful for the help I have received from the library, especially from Ingrid Talman. I would also like to thank Birgitta Olsson and Sören Gustafsson for help with practical matters.

I have been extremely fortunate to work in such a friendly environment. I want to thank Danny Brash, Lisa Brouwers, Maria Croné, Patric Dahlqvist, Petra Fagerberg, Ylva Fernaeus, Karin Hansson, Jesper Holmberg, Fatima Jonsson, Martin Jonsson, Klas Karlgren, Petter Karlström, Peter Lönnqvist, Johan Mattsson, Mona Påhlman, Mats Skoglund, Hillevi Sundholm, and Patrik Werle for the (too) many hours of discussions in the coffee room.

I will also take the opportunity to thank my friends Martin Howard, Martina Leons, Malin Borell, Linda Saveholt, and Inger Södergren for, well, being my friends. I will be a much better person now that this thing is finished!

My deepest thanks go to my one and only Tor Johnson for never-ending patience, laughter, and love. Thanks also for the implementation of Keegle's Web front-end.

Although I am convinced that they will never read it, this dissertation is dedicated to my two sisters, Anna and Sara.

Table of contents

Chapter 1

Introduction

This dissertation shows that automatic keyword extraction augmented with linguistic knowledge outperforms approaches based entirely on standard statistical methods.

1.1 Motivation

In order to make further progress in the field of automatic text retrieval, approaches extending current standard full text indexing methods need to be investigated. This dissertation advocates keyword indexing as one such feasible approach, where *keywords* refers to a small set of terms reflecting the content of a single document. Keywords can play an important role in supporting a user in the search for relevant information. They may serve as a dense summary for a document, they can lead to an improved performance of a search engine, or they may constitute the input to a clustering mechanism, to give just some examples. Most documents lack keywords, and since it would be much too time-consuming, expensive, as well as tedious to manually assign keywords for all these documents, automatic methods must be explored.

Automatic keyword indexing is a research topic that has received less attention than it deserves, considering the potential usefulness of keywords. As is the case with most attempts to automate tasks related to human cognition, automatic keyword indexing is not trivial. A prerequisite for automating any human task is that it is not a random act. That keyword indexing complies with this requirement is shown by the fact that humans in general are able to select keywords for a given text, in a non-random fashion. The research presented in this dissertation builds on the assumption that keywords have certain properties that distinguish them from non-keywords. Although a chosen set of keywords can be expected to differ from a set selected by another person or from a set selected by the same person at a different time, there is still likely to be an agreement on some of the words, and — just as importantly — on which words that are *not* keywords. A word being a suitable keyword for one document is probably inappropriate for another document. Also, humans are to a certain extent able to select keywords for unfamiliar documents, and possibly even able to appropriately select unfamiliar words. Thus, it can be assumed that the properties making a word a keyword are found on another level than the word's semantics (the meaning of a word). The challenge is then to determine what these properties are.

1.2 PROBLEM

The work discussed concerns automatic keyword *extraction*. That the keywords are extracted means that the selected terms are present verbatim in the document. Since the word *word* is rather vague (is, for example, *compact disc player* one word or is it three words?) *token* will be the preferred term throughout the dissertation, where a token is a sequence of characters delimited by space. A keyword may consist of one or several tokens. In addition, a keyword may well be a whole expression or phrase, such as *snakes and ladders*. Whereas humans usually are good at knowing when a number of consecutive tokens should be treated as a unit (as one "word" or as one phrase), this is not evident for an automatic indexer. Therefore, the first problem when automating the task is to determine where a keyword begins and ends in a running text. In the dissertation, this procedure is referred to as defining the *candidate terms*.

In the presented research, three different approaches to select candidate terms from written documents are defined and evaluated. One term selection approach is statistically oriented. This approach extracts all uni-, bi-, and trigrams from a document. The two other approaches are of a more linguistic character, utilising the words' parts-of-speech (PoS), that is, the word class assigned to a word. One approach extracts all noun phrase (NP) chunks, and the other all terms matching any of a set of empirically defined PoS patterns (frequently occurring patterns of manual keywords).

Once the candidate terms have been extracted, it is likely that too many terms have been selected to make them useful as keywords. A filtering of terms is achieved by constructing a model that — based on the values of the defined features — can make this distinction for the extracted candidate terms. In the experiments presented in this dissertation, four features are selected and evaluated for filtering candidate terms. As is the case with the term selection approaches, both statistical and linguistic measures are explored. The statistical measures investigated are *term frequency* (how often a candidate term occurs in a document), *inverse document frequency* (in how many documents a candidate term occurs in relation to the total number of documents in a collection), and the position of the first occurrence of the candidate term. The linguistic feature is (again) based on the candidate term's part(s)-of-speech.

1.3 METHOD

In the research presented in this dissertation, the approach taken to automate keyword extraction is that of *supervised machine learning*, that is, *prediction models* are constructed from documents with known keywords. Before any learning can take place, the data that the learning algorithm will be exposed to must be pre-processed and represented in a form that the algorithm can understand. In the case of automatic keyword extraction, the pre-processing consists of two steps. The first step is to extract the candidate terms from the documents. The second step is to calculate the feature values for the candidate terms. The goal for the learning is then to find the feature values discriminating the candidate terms that are appropriate

keywords from those candidate terms that are inappropriate. To train the prediction models, the feature values of the known keywords are used.

To evaluate the performance of the constructed prediction models, these are applied to previously unseen (by the algorithm) documents, for which keywords are selected. For the evaluations presented in this dissertation, keywords assigned by professional indexers to these unseen documents are used as a gold standard, that is, the manual keywords are treated as the one and only truth. The measures used for the evaluations are *precision* (how many of the automatically assigned keywords that are also manually assigned keywords) and *recall* (how many of the manually assigned keywords that are found by the automatic indexer). These two measures may be combined in the *F-measure*, which is the third measure used for the evaluations.

1.4 CONTRIBUTION

The aim of the presented research is to develop an algorithm for automatic keyword extraction. In this dissertation, a number of experiments are described, where the performance of the algorithm is incrementally improved. Also, an implementation of the best performing algorithm is presented, in which the automatically assigned keywords are used to give more information about the results returned by a Web search engine. The research is linguistically oriented in the sense that the output from natural language processing (NLP) tools is a considerable factor both for the pre-processing of the data, as well as for the performance of the prediction models. The keyword extraction is developed for English, but many ideas put forth in this dissertation are applicable to other languages as well.

Below, the main contributions of the research presented in this dissertation are summarised:

- An empirical comparison of one statistical and two linguistically oriented approaches to extract candidate terms, showing that one of the linguistic methods (the NP-chunk approach) outperforms the statistical method (Hulth, 2003a; Hulth, 2003b; Hulth, 2004b).
- An empirical investigation showing that when part-of-speech information is used as a feature, the performance of the prediction models is improved (Hulth, 2003b).
- An empirical investigation showing that when combining models built from different representations, an improved performance is achieved. The representations differ in how the candidate terms are extracted from the documents (Hulth, 2004a).
- An empirical investigation showing that when using regression (the output is a numeric value between zero and one) instead of classification (the output is binary) for predicting the keywords, better performance is achieved (Hulth, 2004b).
- An algorithm for automatic keyword extraction combining statistical and linguistic methods.

1.5 OUTLINE

In Chapter 1 (this chapter), the motivation for the research is given, as is a description of the contributions of the work. In Chapter 2, the task of indexing is put into context, and full text indexing as well as manual keyword indexing are discussed. This is followed by an overview of previous work on automatic indexing. In Chapter 3, the research problem and the experimental methodology are described. Also, problems concerning the evaluation of automatically selected keywords are discussed.

Chapters 4–8 contain the bulk of the empirical research. In more detail, Chapter 4 investigates the different representations of the data, that is,

how the candidate terms are defined and which features are calculated. In Chapter 5, a number of prediction models are evaluated. The models are based on the three term selection approaches using three and four features respectively. In Chapter 6, experiments on how to improve the performance by combining prediction models are presented. In Chapter 7, three minor changes to the keyword extraction are suggested. One concerns how the candidate terms are extracted by the NP-chunk approach, the second concerns how the inverse document frequency is calculated, while the third change concerns how the weights are set to the positive examples. The chapter also presents one major change, namely experiments in which the learning task is redefined, by applying regression instead of classification. Thereafter, an example of the extraction is given. Chapter 8 describes an implementation of the best performing algorithm, in which automatically extracted keywords are integrated with the results from a Web search engine.

The last chapter (Chapter 9) contains three sections. In the first section, the algorithm for automatic keyword extraction is outlined. The second section gives directions for future work, and the third and last section concludes this dissertation.

Chapter 2

Indexing methods

In order for a document to be retrievable with respect to its content rather than through the title or the author — be it a physical or a digital document — the content must be analysed and represented. This process of representation is referred to as *indexing*[1]. How the indexing is done is crucial, since it sets the limits for how successful the future retrieval can be. According to Salton and McGill (1983), indexing is "probably the most difficult [...] operation required in information retrieval". There are several types of systems for representing documents. To begin with, almost every document has a *title*, and many documents also have an *abstract*. Further, an individual document can be given a *table of contents* that points to certain passages, based on the *headlines* in the document. A *back-of-book index* has the same purpose as a table of contents, but is more often associated with physical documents. It is also, in general, more detailed than a table of contents. Another example of a representation system for text — digital or physical — is *indexing by topic*: each document is assigned one or more labels (or categories) reflecting its content. Documents obtaining the same label are assumed to deal with the same topic and may be clustered together. Physical documents can be placed on the same shelf, while for example the titles of digital documents can be displayed together. In contrast to categorisation or classification, *subject indexing* describes the content of

[1] The term *index* is derived from a Greek word meaning *to point*.

a document in more detail, and several terms or *keywords* are assigned to each document. According to Baeza-Yates and Ribeiro-Neto (1999) keywords — whether manually or automatically assigned — reflect the "logical view of the document". It is, however, not possible to exactly demarcate categorisation and subject indexing. Most of today's search engines — that retrieve digital documents — work on indices that are based on statistical analyses of the documents in the collection at hand. It is easy to let a computer create an index of all words present in a document, so called *full text indexing*.

2.1 MANUAL KEYWORD INDEXING

Before there were computers that could easily process large amounts of documents, the prevailing method for indexing for future information needs was manual subject indexing. This was — and still is — usually done by professional indexers. The keywords may either be selected from a limited set — such as a thesaurus — in which case the task is referred to as *keyword assignment* or *controlled indexing*. Or, they can be any suitable terms, in which case the task is called *uncontrolled* or *free indexing*. If the keywords that are selected for the free indexing are chosen from the document at hand, the task is referred to as *keyword extraction* (Lancaster, 1998).

The purpose of the subject indexing is to describe what a document is about — to describe "the inherent subject of a document" (Hodge, 1992). Hodge further states that the subject of a document should be separated from the "meaning" of a document. The reason why these two notions should be kept separate is, according to Hodge, that the meaning of a document varies between sessions, tasks, and users: What is useful at one time may turn out to be completely inadequate at another time, as neither humans nor their interpretation of information is static (Brown and Duguid, 2000). The meaning depends, among other things, on a user's background knowledge and current information needs. In contrast to the meaning, Hodge claims that it is possible to objectively find and express the subject of a document. This idea is also discussed by Langridge (1989), who points out the importance of not confusing what a document is "about" with what it is "for". Langridge continues by stating that although impor-

tant, these two notions are often confused, proven by the fact that not even professional indexers are consistent. As reported by, for example, Dexter and Zunde (1969), the indexing consistency between two indexers varies between 20 and 80%. Langridge claims that such an inconsistency is due to the fact that the task of indexing is difficult, not because it is impossible. One advantage with automatic indexing, compared to manual indexing, is the consistency of the performance.

O'Connor (1996) has listed a set of questions that should be considered when manually indexing a document:

- How many terms should be extracted?
- Which terms should be extracted?
- Should the terms be derived rather than extracted?
- In what order should they be presented?
- Should concepts be generalised?
- What rules should guide the extraction?

There are no definite answers to these questions (neither for manual nor for automatic keyword indexing). O'Connor concludes that at least the first three questions are best answered by the user, and in addition that the rules of extraction in general are implicit. In most applications it is, however, impractical to involve a user at the stage of keyword indexing. For this reason, the set of selected keywords should describe the content of the document as generically as possible, and be useful for more than one person at a specific occasion. In other words, the goal should be to find the keywords describing what a document is "about".

2.2 AUTOMATIC INDEXING

In this section, a number of methods to automate the task of indexing for future retrieval will be discussed. The dominant method today is full text indexing, which is how most search engines work. Many of the present

ideas about automatic text retrieval were first put forth in the late 1950s (Luhn, 1957; Luhn, 1959). Even if some search engines work in a manner more sophisticated than the one described below, the underlying ideas have remained essentially the same. The two basic measures used for full text indexing are *term frequency* (a term's frequency of occurrence in a document), and *inverse document frequency* — the importance of a term is inversely proportional to the number of documents in which it occurs (Sparck Jones, 1972). Words that are so common that they have no discriminating ability may be stored in a *stop list* and removed from the texts before the index is created (for example determiners such as *the*, and conjunctions such as *and*). A common way to merge different forms is *stemming*, that is, removing affixes to achieve the common root. For example, *programming*, *programmer*, *programmers* and *program* may all be represented as *program**, where the asterisk means any ending. (This process must, however, be more elaborate for language types morphologically richer than English.) As the created index will not be shown to the users, the representation may be more or less meaningless for humans, as long as it makes sense to the search engine. For the same reason, it is not a problem that an index for a collection is vast; it is nothing but an internal representation. A user who wishes to find documents about a certain topic states the information need to the search engine — usually by writing one or more terms expected to be present in the relevant documents. This request is processed by the search engine in the same way as the documents, and the *query* resulting from this process is thereafter matched to the full text index. Finally, a list of presumably relevant documents is returned to the user.

Although much has been achieved in the area of document retrieval, there is no doubt that enhancements to present retrieval tools are more than welcome (Tzoukermann *et al.*, 2003). One drawback with full text indexing is its limitation in that the index is only useful for the search engine: it will make little sense if displayed to a user, as the user will get a list of almost all words in a document, and in addition some of the terms will not be proper words. Another more problematic aspect is that it is not unusual that thousands of hits are returned to a request submitted to a search engine. There are a number of reasons for why too many documents are retrieved and, more importantly, why too many of these are irrelevant, while many relevant documents are left out. Some reasons for unsatisfactory retrieval results are inherent in human languages. Words and strings

can be polysemous (have multiple meanings), and the same meaning may be expressed by synonyms (words with a similar meaning). We also interpret words differently depending on the context as well as on our personal preferences. Many words present in a text are also likely to be insignificant to the major theme or themes of a document, hence leading the retrieval astray. A request posed by a user may be vague, or the user may not — for some other reason — be able to formulate the information need in a way so that the search engine can return the required documents. This is a shortcoming of the search engine, and not of the user. In case a search engine had any doubts of what the user intended with a request, it should try to find out more about the user's need by, for example, initiating a dialogue with the user. In addition, a user's information need may change during a session, depending on which documents are returned by the search engine.

There are not yet any convincing results that linguistic knowledge improves traditional information retrieval, although it may be useful and even necessary for types of applications other than full text indexing (Sparck Jones, 1999). Sparck Jones lists three such domains: *information extraction*, where templates are used to extract facts about certain events, entities, and relations (see, for example, Gaizauskas and Wilks (1998)); *question-answering*, where the answer to a question posed in natural language is retrieved instead of documents (see, for example, Harabagiu and Moldovan (2003)); and *automatic summarisation* (see, for example, Radev *et al.* (2002)). As pointed out by Jacquemin *et al.* (2002), apart from these three domains, there are probably also other information retrieval tasks that can benefit from natural language processing techniques.

Finding appropriate index terms by means of part-of-speech patterns is a common approach. When inspecting manually assigned index terms, the vast majority turns out to be noun phrases containing a head noun, with zero or more adjectives. According to Carroll and Roeloffs (1969), "The concepts an author is trying to convey is reflected in the number of times non-common adjectives and nouns are used". When PoS patterns are used to extract candidate terms, the problem lies in how to restrict the number of terms, and only keep the ones that are relevant. The most explored feature for filtering the candidate terms is frequency (or occurrence). Barker and Cornacchia (2000) discuss an algorithm where the number of words and the frequency of a noun phrase,

as well as the frequency of the head noun are used to determine what terms are keywords. An extraction system called *LinkIT* (see for example Evans *et al.* (2000)) compiles the noun phrases, and then ranks these according to the heads' frequency. Another example is Daille *et al.* (1994) who apply statistical filters on extracted noun phrases. In that study it is concluded that term frequency is the best filter candidate of the scores investigated. Justeson and Katz (1995) have shown that technical terminology consists mostly of multi-word terms being noun phrases with nouns, adjectives or the preposition *of* (more than 99 per cent), and that single-word terms are rarely of a terminological character.

Earl (1970) discusses a method for automatic back-of-book indexing, where noun phrases are extracted and filtered according to frequency. Lahtinen (2000) has also performed experiments on an automatic back-of-book indexer using linguistic information. Lahtinen concludes that the five most common term combinations are all noun phrases, containing nouns, adjectives, or the preposition *of*. In Lahtinen's experiments not only the occurrence, but also the location of the terms is explored. Terms occurring in, for example, headlines and in sentences at certain positions, such as in the first and last sentences of paragraphs, are shown to contain more relevant index terms. This finding is similar to that of Edmundson (1969), who explored position in a document as a feature reflecting textual importance in early experiments on automatic text summarisation. Boguraev and Kennedy (1999) extract technical terms based on the noun phrase patterns suggested by Justeson and Katz (1995); these terms are then the basis for a headline-like characterisation of a document. Automatic headline assignment is also the goal for a set of experiments presented in Kraaij *et al.* (2002). In those experiments, a system for multi-document summarisation is extended to extract the most informative noun phrase for each document cluster (a set of documents on the same subject), which is then used as the cluster's headline.

Basili *et al.* (2002) have performed experiments on text classification, where better performance is achieved for some of the categories, when each extracted term also contains its part-of-speech tag. Thus, words having different readings are not collapsed, keeping for example the verb *rate* and the noun *rate* as two distinct terms. Including part-of-speech information as a feature for constructing prediction models has been done for headline

generation (Banko *et al.*, 2000). In those experiments it is shown that when the PoS feature is used in combination with positional information, the performance of the models is improved.

A research area that is related to automatic keyword indexing is that of terminology extraction, where all terms describing a domain are to be extracted (see for example Bourigault *et al.* (2001)). As has been pointed out by Kageura *et al.* (1998), among others, the difference between keyword indexing and terminology extraction has not been clearly defined. The aim of automatic keyword indexing is to find a small set of terms that describes a specific document, independently of the domain it belongs to. Thus, the set of terms must be limited to contain only the most salient ones. Automatic keyword indexing may, however, very well benefit from the results of automatic terminology extraction, as appropriate keywords often share characteristics with terms.

To treat automatic keyword extraction as a supervised machine learning task is an approach first proposed by Turney (2000). Machine learning is a research area studying the development of methods for automatic detection of recurring patterns in large data sets. The learning results in prediction models, which are used to predict future instances of the phenomenon under investigation. (For an introduction to the field, see for example Witten and Frank (2000) or Mitchell (1997).) That the learning is supervised means that a number of instances with a known prediction value or class are used to train the models. In the case of automatic keyword extraction, a trained model is subsequently applied to documents for which no keywords are assigned: each defined term from these documents is classified either as a keyword or a non-keyword; or — if a probabilistic model is used — the probability of the defined term being a keyword is given. Applying supervised machine learning to automate keyword extraction is also the approach taken in the work described in this dissertation.

In the experiments described by Turney (2000), a genetic algorithm tuned to the task is compared to a general-purpose algorithm (bagged C4.5 for inducing decision trees). Turney concludes that the genetic algorithm performs better than the decision trees, as measured by the precision of the selected keywords. In those experiments, the candidate terms are all stemmed uni-, bi-, and trigrams that do not have any intervening stop

words. The parameters chosen for the genetic algorithm are, for example, term frequency, position of the first occurrence, whether the term ends with an adjective as judged by its unstemmed suffix, or if it is a verb (if it is included in a list of common verbs). However, these parameters differ slightly from the features used by the decision trees, as does the post-processing. Thus, the difference in the performance might be due to this fact rather than to the performance of the learning algorithms. Before settling on the features used by the decision trees, Turney evaluated 110 different features.

Frank *et al.* (1999) report experiments on a keyword extraction algorithm based on naive[2] Bayes which performs as well as Turney's genetic algorithm, when evaluated on part of Turney's test material. The candidate terms are selected in a manner similar to Turney's, while the set of features used is smaller and simpler. These are the term frequency combined with the inverse document frequency (TF*IDF), and the relative position of the first occurrence of each extracted n-gram. Frank *et al.* also introduce a fourth feature which significantly improves the algorithm, when trained and tested on domain-specific documents. This feature is the number of times a term is assigned as a keyword to other documents in the collection used for training the algorithm.

It should be noted that the performance of the state-of-the-art keyword extraction is much lower than for many other NLP-tasks, such as tagging and parsing, and there is plenty of room for improvements. To give an idea of this, the results obtained by the genetic algorithm trained by Turney (2000), and the naive Bayes approach by Frank *et al.* (1999) are presented in Table 2.1. The number of terms assigned must be explicitly limited by the user for these algorithms. Both Turney and Frank *et al.* report the precision for five and fifteen keywords per document. Recall is not reported in their studies. In the table, their results when training and testing on journal articles are shown, and the highest values for the two algorithms are presented.

In the case of professional indexing, the keywords are normally limited to a domain-specific thesaurus, but not only to those present in the document to which they are assigned. Experiments on how thesaurus descrip-

[2]Making the assumption that the effect of a feature value on a given class is independent of the values of other features.

Table 2.1: *Precision, and the mean value of correct terms for Turney (2000)* and Frank* et al. *(1999)**, for five and fifteen extracted terms.*

	Precision	**Correct**
5 terms*	29.0	1.45
15 terms**	18.3	2.75

tors may be automatically assigned have been performed by, for example, Pouliquen *et al.* (2003), Montejo Ráez (2002), and Ferber (1997). In this dissertation, however, the concern is not to restrict the keywords to a set of allowed terms, as the indexing would be too confined. Also, if an automatic method is to be applicable to documents outside a specific domain, the method must not require a domain-specific thesaurus.

The automation of keyword indexing can be done to different degrees. There are intermediate steps between fully manual and fully automatic approaches. Hodge (1992) refers to such a spectrum of possible activities as the "automated support continuum". By giving computer support to professional indexers, consistency as well as productivity can be raised. An example of experiments on semi-automatic indexing is presented in Hulth *et al.* (2001a). In those experiments, the aim was to build a keyword indexer that found all manually assigned keywords as well as a number of additional keywords, as evaluated on previously manually indexed documents. The experiments were performed on bills written in Swedish from the Swedish parliament, using *inductive logic programming* (see for example Nienhuys-Cheng and de Wolf (1997)). When knowledge from a domain specific thesaurus is included — by utilising the hierarchical organisation of the terms, such as *broader term* and *narrower term*[3] — the performance of the keyword indexer is improved.

Although manual indexing at a first glance may appear to be rather routines work, it is still a fairly complex task to automate. In addition, the manual effort put into an indexing task should not be underestimated: "The unwise seem to be of opinion that any fool can index, but [...] the wise think differently" (Wheatley, 1878).

[3]A broader term is a more general term (a hyperonym), and a narrower term is a more specific term (a hyponym).

Chapter 3

Problem and research approach

The aim of the research presented in this dissertation is to define an algorithm for automatic keyword extraction. This chapter presents the research problem and the experimental methodology in more detail.

3.1 Research problem

As discussed in the introduction, the task of automatically extracting keywords can be divided into two subtasks:

- Extract candidate terms
- Filter the extracted candidate terms in a fashion that retains those that are appropriate keywords and rejects those that are inappropriate.

In Section 2.2, the experiments on automatic keyword extraction performed by Turney (2000) and Frank *et al.* (1999) were introduced. Their

approach to select the candidate terms is by means of extracting n-grams, where n equals one, two, and three. The problem when extracting uni-, bi-, and trigrams is that the length of the candidate terms — per definition — is limited to three tokens. In the data set (described in Section 3.2) used for training the prediction models evaluated in this dissertation, 9.1% of the manually assigned keywords consist of four tokens or more, and the longest keywords have eight tokens[1] (see Table 3.1). As can be seen in this table, the majority of the manual keywords contain two tokens.

Table 3.1: *The number of tokens in the manual keywords assigned to the training set (in total and in per cent).*

Tokens	**# of keywords**	**% of keywords**
1	1 344	13.7
2	5 065	51.8
3	2 488	25.4
4	673	6.9
5	157	1.6
6	43	0.4
7	14	0.1
8	3	0.0

As also discussed in Section 2.2, extracting candidate terms by means of linguistic criteria is a common approach. When inspecting manually assigned keywords, the majority turns out to be noun phrases (where the noun phrase may consist of only one noun). This insight is the basis for the criterion. Extracting noun phrases means that the length of the candidate terms is not something arbitrary, but reflects a linguistic property that is valid for keywords.

In the experiments described in this dissertation, three different term selection approaches (that is, approaches to extract candidate terms) are defined and evaluated. The first one is similar to the approach suggested by Turney (2000) and Frank *et al.* (1999), that is, extracting uni-, bi-, and trigrams, removing those that begin or end with a stop word. This approach

[1] An example of a keyword in the training data containing eight tokens is *National Information Clearinghouse on Children who are Deaf-Blind.*

is statistical in nature. The two other approaches are more linguistically oriented. The first linguistic approach is to extract all noun phrases in the documents, as judged by a NP-chunker (a tool that automatically marks syntactic boundaries based on assigned PoS tags). The second linguistic term selection approach explored is based on a number of defined part-of-speech tag sequences (or patterns). It extracts all words or sequences of words matching any of these, relying on a PoS tagger (a tool that automatically assigns each token's part-of-speech). The patterns correspond to the PoS tag or tags assigned to the manual keywords present in its abstract in the training data, provided that the pattern occurs ten or more times. This approach still retains the idea of keywords having certain linguistic properties, but is based on empirical evidence in the data. To assign the PoS tags and the NP-chunks, *LT CHUNK* from the Language Technology Group at the University of Edinburgh is used[2].

Regardless of term selection approach, the majority of the extracted candidate terms are not keywords (although the three approaches perform differently). The decision of what feature values that discriminate candidate terms that are appropriate keywords from those that are not is handed over to a machine learning algorithm. This has the advantage that a large amount of data can be investigated, and thus more reliable models can be constructed. Four features are selected: the three domain-independent features[3] proposed by Frank *et al.* (1999), that is, term frequency, IDF, and relative position of first occurrence; also a feature previously unexplored for this purpose, namely the PoS tag(s) assigned to the term is investigated. Initially, experiments with both unstemmed and stemmed terms are performed, using Porter's stemmer (Porter, 1980). Later, only stemmed terms are used since these result in better performance on the separate validation data.

In Chapter 4 to Chapter 7 the experiments leading to the keyword extraction algorithm are described. These experiments can be seen as a process in which the keyword extraction algorithm is incrementally developed.

[2]Available at `http://www.ltg.ed.ac.uk/software/pos/index.html`.

[3]The fourth feature proposed by Frank *et al.* (1999) (see page 14) is not investigated in this dissertation, as it is only applicable if the keyword extraction is domain-specific.

The work can be summarised as follows:

- The extraction capability of each one of the three defined term selection approaches is investigated, as is the discriminative power of the four features (Chapter 4).
- A number of prediction models are constructed and evaluated. The models are based on the three term selection approaches using three and four features respectively (Chapter 5).
- Experiments on how to improve the performance by combining prediction models are performed (Chapter 6).
- Three minor changes to the keyword extraction are empirically verified. One concerns how the candidate terms are extracted by the NP-chunk approach, the second concerns how the inverse document frequency is calculated, while the third change concerns how the weights are set to the positive examples (Chapter 7).
- The learning task is redefined, by applying regression instead of classification for the predictions (Chapter 7).

3.2 EXPERIMENTAL DATA

The collection used for the experiments described in this dissertation consists of 2 000 abstracts in English, with their corresponding title and keywords from the Inspec database[4]. The abstracts are from the years 1998 to 2002, from scientific journal papers, and from the disciplines *Computers and Control*, and *Information Technology*. Each abstract has two sets of keywords — assigned by a professional indexer — associated to them: a set of controlled terms (keywords restricted to the Inspec thesaurus); and a set of uncontrolled terms that can be any suitable terms. Both the controlled terms and the uncontrolled terms may or may not be present in the abstracts. However, the indexers had access to the full-length documents when assigning the keywords, and not only to the abstracts. Since the approach taken to the keyword indexing is extraction, only the uncontrolled

[4] `http://www.iee.org/publish/inspec/`

terms are considered in the experiments presented in this dissertation. The reason is that these to a larger extent are present in the abstracts (76.2% as opposed to 18.1% for the controlled terms).

The set of abstracts is arbitrarily divided into three sets: a training set consisting of 1 000 documents (to construct the prediction models), a validation set consisting of 500 documents (to evaluate the models, and select the best performing one), and a test set with the remaining 500 abstracts (to get unbiased results). The sets of manually assigned keywords are removed before the candidate terms are extracted. For all experiments the same training, validation, and test sets are used. In the experiments — when tuning the parameters for the learning, or modifying the keyword extraction algorithm — the model that has the best performance on the validation data is chosen. The results presented in the dissertation are all from the test data. In other words, the results presented in the dissertation show how well the algorithm can be expected to perform on unseen data.

3.3 Learning prediction models

The machine learning method used in the experiments is an implementation of *rule induction*, where the prediction models constructed from the given examples consist of a set of rules. The system applied is called *Rule Discovery System* (RDS, 2003). The strategy used to construct the rules is *recursive partitioning* (or *divide-and-conquer*), which has as its goal to maximise the separation of the classes for each rule. The resulting rules are hierarchically organised (that is, as decision trees). In order to construct rules that are not over-fitted — to avoid rules that are too closely tuned to the training data and will generalise poorly — half of the training data are saved for pruning the rules. This action is part of how the learning algorithm is implemented in RDS.

The machine learning system used allows for different ensemble techniques to be applied, meaning that a number of models are generated and then combined to make the prediction. The one used for the experiments described in Chapter 5 is *bagging*[5] (Breiman, 1996). In bagging, examples

[5] From the term *bootstrap aggregation*.

from the training data are drawn randomly with replacement until a set of the original size is obtained. This new set is then used to train a classifier. This procedure is repeated n times to generate n classifiers, for which a majority vote is taken to classify an instance.

In all but the last set of experiments, the task is treated as a classification task, that is, the output from the machine learning algorithm is categorical (a candidate term is either a keyword or not). Consequently, the system itself limits the number of extracted keywords per document. In the last set of experiments, the task is treated a regression task (Breiman *et al.*, 1984), where the output is a numerical value. For these runs, the output is a real value between zero and one, intended to reflect the likelihood that a candidate term is a keyword. To limit the number of keywords selected per document, a threshold value is empirically set. The experiments on regression show that this method leads to better performance of the models.

The input to the learning algorithm consists of so called *examples*, where an example refers to the feature value vector for each, in this case, candidate term. For the classification runs, the examples corresponding to manual keywords are assigned the class *positive*, and the others are given the class *negative*. For the regression runs, an example corresponding to a manually assigned keyword is given the value one, while all other candidate terms are given the value zero. Figure 3.1 shows three feature value vectors from the training data, one for each term selection approach. Once the feature values have been calculated, the term itself serves only as an identifier. In Figure 3.1, the identifier contains the document number and the candidate term.

Since the data set is unbalanced, that is, there is a larger number of negative examples than positive examples, the weight given to the positive examples may be adjusted. In total, there are a number of parameters to adjust in all but infinitely many ways. For this reason, it is not possible to exhaustively explore all alternative settings, and to guarantee that the optimal parameter setting is found.

It should be noted that the intention with this dissertation is not to argue for a certain machine learning approach in favour of any other. However, one advantage with rule induction is that the rules may be inspected and are easily interpreted, and thus might give an insight into how the learning

Identifier	TF	IDF	POS	TAG	Class
1001_theories_are_false	1	2.30	0.83	NNS_VBP_RB	Negative
950_classical_chaos	1	2.30	0.21	JJ_NN	Positive
1013_controller	7	0.33	0.03	NN	Negative

Figure 3.1: *Feature value vectors for three candidate terms.* TF *stands for term frequency, and* POS *for relative first position. As for the PoS tags* (TAG), NNS *denotes a plural noun,* VBP *a verb present tense,* RB *an adverb,* JJ *an adjective, and* NN *a singular or a mass noun.*

component makes its predictions. In Figure 3.2, two rules from one of the keyword extraction models are shown. On the other hand, the interpretation is more difficult when applying ensemble techniques. The number of rules also limits of how much information a prediction model may convey to a human.

```
if IDF<0.15 and TAG!=NNP then Negative

if TF>3.5 and TF<7 and IDF>0.525 and IDF<0.915
and POS<0.205 and TAG!=DT_NN then Positive
```

Figure 3.2: *An example of two rules from the best performing classification model for the NP-chunk approach.* TF *stands for term frequency,* POS *for relative first position, and* != *for not equal.* NNP *refers to a proper noun, and* DT_NN *refers to determiner noun.*

3.4 EVALUATION

An important phase of any research process is the evaluation phase. In this section, five possible settings for evaluating automatically extracted keywords will be discussed. As stated in Section 2.1, keyword indexing

is a difficult task. The goal for the indexer (human or automatic) should be to find the set of keywords describing what the document is about (its subject), and not what it is for (that is, the meaning that varies depending on the situation in which a document is retrieved). However, it is difficult to see how the notions of what a document is about and what it is for could be separated. Even if there were such a thing as a set of keywords exactly describing what a document is about — valid for anybody at anytime — there is seemingly no way to unveil if this is really the case: the chosen keywords cannot be objectively judged nor can we say what requests users may pose in the future. Therefore the idea of an objectively correct set is discarded, and it is assumed that a preferred set of keywords varies between individuals as well as sessions. As a consequence, evaluating keywords is not trivial. The only way that the suitability of a set of keywords can be judged, is if a user with a specific need at a specific time gives the judgement. This is, of course, not a realistic scenario for most evaluations, and alternatives must be found. (However, this does not mean that the aim of the indexing should not be to find a set that as well as possible reflects the content in a generic way.) This problematic characteristic of evaluation is something that automatic keyword indexing shares with for example automatic text summarisation (perhaps an even more difficult task), where there is no objectively best summary.

Evaluations of natural language processing systems can be either extrinsic or intrinsic (Sparck Jones and Galliers, 1996). An extrinsic evaluation means that the quality of keywords is evaluated in relation to how well they support a user in solving a specific task. Evaluating the automatically extracted keywords in an intrinsic manner would mean to measure the acceptability of the keywords. Another dimension is whether human judges are involved in the actual evaluation or if the evaluation is performed automatically, reusing previously given human judgements.

An example of an extrinsic evaluation in which human assessors participate at the time of evaluation would be to ask a number of users to solve a set of specific search tasks using a search engine. Keywords extracted automatically from the search hits could be displayed on the result page, together with the title of the document from which they were extracted. To evaluate the function of these keywords — serving as a dense summary for each document in the hit list — the users would be asked to judge

how satisfied they were with their results, and how sure they were that they had found the "right" documents. These judgements could then be compared to judgements given by users who have no keywords displayed. Another measure could be the time difference for finding relevant information between users having keywords displayed and those without. To get an upper-bound for the performance, the same search tasks could be solved with manually assigned keywords displayed. There are two drawbacks with this type of evaluation. Firstly, it requires users, something which does not easily allow for large-scale quantitative evaluations. Secondly, it requires an implemented and fully functioning automatic keyword extractor. Thus, it is not a feasible approach during the development phase.

An extrinsic evaluation that could be performed automatically is proposed in Hulth (2001). The method is based on the assumption that an information retrieval search in a collection where each document is represented by a set of appropriate keywords, would result in higher precision than when performing the same search in an index created from the full text documents. The method requires a set of queries and documents for which relevance judgements exist (or may be provided). In Hulth (2001), a small-scale evaluation to examine the potential of this type of evaluation is presented. About 15 000 documents originating from the TREC[6] (Text REtrieval Conference) collection, each having between one and twelve professionally assigned keywords, are used in the experiment. The results from nine queries are evaluated. For three of these queries, the precision increases as does the recall, when submitting the query to the keyword index compared to submitting the query to the full text index. In four cases, the precision increases, while the recall decreases. For one query, the precision decreases, while the recall increases, and for one query both the precision and the recall decrease. The presented experiment is, as mentioned, on a small-scale, but points to the possibility of an extrinsic and automatic evaluation of automatically selected keywords.

For automatic summarisation, the same idea has been put forth by Brandow *et al.* (1995), who also performed a small-scale experiment with twelve queries. Their results show that searching in an index created from automatic summaries leads to an increase in precision, but that this also causes a dramatic decrease in recall. The drawback with this type of extrin-

[6] `http://trec.nist.gov/`

sic evaluation is that it requires queries and relevance-judged documents for these queries. An approach based on the same idea is explored in large-scale by Radev *et al.* (2003) with a new measure called *relevance correlation*. The assumption is that the less the performance of the retrieval decreases, the better is the summariser. Relevance correlation measures the correlation between the ranking scores of the results returned by a query submitted to a full text index and those returned by the same query submitted to a summary index. It is also assumed that the relevance scores provided by the search engine are reliable when it comes to the documents' relevance (something that is not always the case). This method should be just as applicable for evaluations of keywords.

An example of an intrinsic evaluation of a keyword extraction system involving humans at the time of evaluation is given by Turney (2000). Users submitted URL:s to the system, and were then presented with a number of keywords for each Web page. The user then gave the opinion of whether these keywords were "good" or "bad". The results show that the acceptability of the automatically extracted keywords is much higher than the precision calculated from the manually assigned keywords. This type of evaluation, however, does not account for the silence of the system, that is, what suitable keywords that the system might have missed. This could be solved by asking human judges to also mark additional keywords from the documents. However, as mentioned previously, involving users at the time of evaluation does not easily allow for large-scale quantitative evaluations.

An intrinsic evaluation that may be performed automatically, is to apply a keyword extraction algorithm to documents that already have manual keywords assigned. Using manually constructed keys for evaluation is done in many natural language processing tasks. One example where this type of evaluation is used, and which is close to keyword indexing, is automatic text summarisation. This is, for example, the case for the annual document understanding conference[7] (DUC); a "competition" in automatic summarisation. However, just like humans often disagree on which summary is the best for a certain document, they also often disagree on which keywords that are appropriate. As discussed in Section 2.1, the inter-judgement agreement between professional indexers may be as low as 20%. By instead asking a number of people to manually extract keywords from each document, this

[7] `http://www-nlpir.nist.gov/projects/duc/`

problem could be solved. If several persons selected keywords for the same documents, the majority vote could be taken of which keywords that are acceptable or missed, as to minimise the variability between the human judges. This has been done for automatic summarisation by for example Klavans *et al.* (1998). An alternative could be to calculate the *Kappa* coefficient (Siegel and Castellan, 1988) between the human annotators, and between the human annotators and the system. This has also been used for summarisation by Radev *et al.* (2003). However, getting a large set of manually annotated data is not trivial, especially if the keywords are to be assigned by professional indexers (to guarantee a certain quality and consistency). For this reason, documents having keywords assigned by one professional indexer are used for the evaluations described in this dissertation. The results of the evaluation will nonetheless provide an insight into the quality of the automatically assigned terms, especially so as the manual keywords are assigned by professional indexers. When reusing this type of manually annotated data, the judgements are treated as the one and only truth, that is, they are used as a gold standard. Using manually assigned keywords as the gold standard is also done by Turney (2000) and Frank *et al.* (1999). The advantage of this type of evaluation is that it allows for quick and frequent evaluations.

Two measures used in the dissertation to evaluate the selected keywords are *precision* and *recall*; two common performance measures used, for example, for information retrieval tasks. Precision is — in the case of automatic keyword extraction — the proportion of automatically selected keywords that are also manually assigned keywords:

$$Precision = \frac{\#\ of\ manual\ keywords\ automatically\ selected}{Total\ \#\ of\ automatically\ selected\ terms}$$

Recall is the proportion of manually assigned keywords found by the automatic method:

$$Recall = \frac{\#\ of\ manual\ keywords\ automatically\ selected}{\#\ of\ manual\ keywords\ in\ document}$$

In this dissertation, the precision and the recall values are shown as percentage[8]. To calculate the recall, the number of manually assigned keywords present in the corresponding data set is used. In doing so, the three term selection approaches may be compared, although a varying number of manually assigned keywords are present in the pre-processed data (depending on term selection approach). This figure varies slightly for the unstemmed and the stemmed data, and for the two, the corresponding value is used. In should also be noted that the main concern is the positive examples, and that the number of correctly classified negative examples is not evaluated.

Precision and recall are combined in the *F-measure* — a standard information retrieval metric — defined as:

$$F_\beta = \frac{(\beta^2 + 1) * Precision * Recall}{\beta^2 * Precision + Recall}$$

When $\beta > 1$, precision is favoured, while recall is favoured when $\beta < 1$. As the proportion of correctly suggested keywords is considered equally important as the proportion of keywords assigned by a professional indexer that is detected, β is assigned the value 1 in this work, giving precision and recall equal weights. Thus the formula:

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

is used for calculating the F-measure.

Yet another alternative for an intrinsic evaluation that could be automated, and that would not even require pre-annotated data, would be to compare automatically extracted keywords to those chosen by another system. Due to the complexity of such a comparison, this approach has not been investigated.

To summarise this section, the evaluations of the automatically selected keywords, which are presented in this dissertation are intrinsic, that is, the quality of the assigned keywords is evaluated. For the evaluations,

[8]The proportions are thus multiplied by 100.

keywords assigned by professional indexers are used as a gold standard (where one indexer per document has assigned the keywords). The measures used are precision, recall, and F-measure. It should be noted that the chosen evaluation method, that is, using manually assigned keywords as the gold standard, affects how well the system can be expected to perform as reflected in the figures of the evaluation measures. It is to a certain extent inevitable that keywords selected by a prediction model differ from those selected by a human indexer, as not even professional indexers agree on what set of keywords best describes a document. One reason for this disagreement is that keyword indexing deals with natural language, and humans are inherently individual in such tasks. This, in turn, is because human languages allow for syntactic and semantic variations.

Chapter 4

Pre-processing the data

When preparing the data to be processed by a learning algorithm, there are several important aspects to consider. In this chapter, two aspects that are relevant for automatic keyword extraction are investigated: how the candidate terms are selected from the documents, and what feature values of these candidate terms that are assumed to discriminate keywords from non-keywords.

Parts of the work discussed in the chapter have been presented at the 14th Nordic Conference on Computational Linguistics (NoDaLiDa) held in Reykjavík, Iceland in 2003. The talk was entitled *Analysing term selection approaches and features for automatic keyword extraction* (Hulth, 2003a). Other parts have been discussed in *Improved automatic keyword extraction given more linguistic knowledge*, published in the Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP) held in Sapporo, Japan (Hulth, 2003b).

4.1 Introduction

In a machine learning task, one of the most important issues is how the data that the algorithm is to learn from are represented: what is given as

input to the learning algorithm. When learning to extract keywords, the first problem is to identify the candidate terms in the texts. Since a keyword may consist of one or several tokens, a decision on where a term starts and ends in a running text must be made. The second problem is to define the features of the selected candidate terms, so that the learning algorithm can create a model deciding which candidate terms that are appropriate keywords, and which are not appropriate. This chapter investigates the potential of three term selection approaches by showing how many of the manually assigned keywords each approach is able to find. It also shows the distribution of the keywords and the non-keywords respectively, for each term selection approach, for the values of four selected features.

4.2 Representing the data

This section will describe three approaches to select the candidate terms from the documents, that is, give three different definitions of what word or sequence of words in a written text that has the potential to be a keyword. As mentioned in Section 3.2, the set of 2 000 documents is divided into three sets: a training set of 1 000 documents, a validation set consisting of 500 documents, and the remaining 500 documents are saved for testing. The document set used for investigating the potential of the different representations consists of the 1 000 abstracts in the training set. The length of the abstracts in this set varies from 15 to 512 tokens, and the median is 124 tokens. The total number of tokens in this set is 128 098. The total number of (uncontrolled) keywords assigned to the documents in the set is 9 787. The number of keywords per document is 1 to 30, with a median value of 9 keywords. Of these keywords are 7 459 unstemmed and 7 557 stemmed present in their corresponding abstract.

4.2.1 Defining the candidate terms

The first investigated term selection approach is the same as that proposed by Frank *et al.* (1999), who extract stemmed uni-, bi-, and trigrams, after removing those beginning or ending with a stop word. However, extracting uni-, bi- and trigrams means that the length of a candidate term — per

definition — is limited to three tokens. In addition the selection does not reflect any linguistic properties that a keyword may have. If inspecting manually assigned keywords, the majority turns out to be noun phrases (see also Section 2.2). For this reason, two alternative approaches are examined. These are to extract all noun phrase chunks as determined by a chunker, and to extract all terms matching any of a set of part-of-speech tag sequences. The PoS tag patterns are those PoS tag sequences of the manually assigned keywords present in the abstracts, where the pattern occurs ten or more times in the training set. In total 56 patterns are defined. This approach still captures the idea of keywords having certain syntactic properties, but is based on empirical evidence in the training data. Both unstemmed and stemmed terms are investigated for the linguistically oriented approaches. For the stemming, Porter's stemmer is used (Porter, 1980).

An example of how the three term selection approaches extract candidate terms from the documents will now be given. The sentence:

Iceland is a very beautiful island.

is used to exemplify how each approach works.

The *n*-gram approach

First, all uni-, bi-, and trigrams in the texts are extracted. Thereafter the terms are case-folded (all characters are converted to lower case), and all terms beginning or ending with a stop word (from Fox (1992)) are removed. The remaining candidate terms are then stemmed. In this dissertation, this manner of selecting terms is referred to as *the* n-*gram approach*. The candidate terms for the example sentence would, for the *n*-gram approach, be:

beautiful
beautiful island
iceland
island

The extraction differs from Frank *et al.* (1999) in the following aspects:

- Only non-alphanumeric characters that are not present in any keyword in the training set are removed (keeping for example C++).
- Numbers are removed only if they stand separately (keeping for example 4YourSoul.com).
- Proper nouns are kept.
- The applied stemming and stop list are different.
- The stems are kept even if they appear only once (which is true for 80.0% of the keywords present in the training set).

The NP-chunk approach

In order to extract NP-chunks, the texts are first processed by a chunker (LT Chunk). This way of defining the terms is in this dissertation called *the NP-chunk approach*. The output for the example sentence when processed by the chunker would be:

> <S> [[Iceland_NNP]] ((is_VBZ)) [[a_DT very_JJ beautiful_JJ island_NN]] ._. </S>

The tags follow the Penn tree bank convention (Marcus *et al.*, 1994), so NNP refers to a proper noun, VBZ is a verb third person singular present tense, DT a determiner, JJ is an adjective (thus, 'very' is mistagged), and NN is a singular noun. The NP-chunks are enclosed by brackets, and the verb chunks are enclosed by parentheses.

As with the n-gram approach, the candidate terms are after the extraction case-folded. Extracting the NP-chunks for the example sentence would result in the following two candidate terms:

> a very beautiful island
> iceland

The PoS pattern approach

To extract the words or sequences of words matching any of the PoS tag patterns, the tags assigned by the chunker (or rather the tagger) are used. This way of defining the terms is in this dissertation called *the PoS pattern approach*. The 56 defined PoS patterns are shown in Figure 4.1. The PoS patterns are listed in order of frequency, where the most frequently occurring pattern comes first. Out of the 56 patterns, 51 contain one or more noun tags[1].

After that the candidate terms have been case-folded, the PoS pattern approach would extract the following candidate terms for the example sentence:

beautiful
beautiful island
iceland
iceland is
island
very
very beautiful
very beautiful island

[1] In Table 4.1, 'Proper adjective' refers to adjectives such as *Hamiltonian*, and *Mediterranean*.

Adjective Noun (singular or mass)
Noun (sing. or mass) Noun (sing. or mass)
Adjective Noun (plural)
Noun (sing. or mass) Noun (pl.)
Noun (sing. or mass)
Adjective Noun (sing. or mass) Noun (sing. or mass)
Proper noun (sing.)
Adjective Noun (sing. or mass) Noun (pl.)
Noun (pl.)
Noun (sing. or mass) Noun (sing. or mass) Noun (sing. or mass)
Adjective Adjective Noun (sing. or mass)
Adjective Adjective Noun (pl.)
Proper noun (sing.) Noun (sing. or mass)
Noun (sing. or mass) Noun (sing. or mass) Noun (pl.)
Proper noun (sing.) Noun (pl.)
Proper noun (sing.) Proper noun (sing.)
Proper noun (sing.) Proper noun (sing.) Proper noun (sing.)
Verb (gerund/present participle) Noun (sing. or mass)
Adjective Noun (sing. or mass) Noun (sing. or mass)
Noun (sing. or mass)
Verb (gerund/present participle) Noun (pl.)
Adjective
Adjective Adjective Noun (sing. or mass) Noun (sing. or mass)
Proper noun (sing.) Noun (sing. or mass) Noun (sing. or mass)
Verb (gerund/present participle)
Adjective Noun (sing. or mass) Noun (sing. or mass) Noun (pl.)
Noun (sing. or mass) Verb (gerund/present participle)
Adjective Proper noun (sing.) Noun (sing. or mass)
Adjective Adjective Noun (sing. or mass) Noun (pl.)
Noun (sing. or mass) Noun (sing. or mass) Noun (sing. or mass)
Noun (sing. or mass)
Noun (sing. or mass) Verb (gerund/present participle)
Noun (sing. or mass)
Adjective Proper noun (sing.) Noun (pl.)

Noun (sing. or mass) Adjective Noun (sing. or mass)
Noun (sing. or mass) Preposition/subordinate conjunction Noun (sing. or mass)
Proper noun (sing.) Proper noun (sing.) Proper noun (sing.) Proper noun (sing.)
Adjective Verb (gerund/present participle)
Proper adjective Noun (pl.)
Proper adjective Noun (sing. or mass)
Adverb Adjective Noun (pl.)
Proper noun (sing.) Noun (sing. or mass) Noun (pl.)
Adjective Verb (gerund/present participle) Noun (pl.)
Verb (gerund/present participle) Noun (sing. or mass) Noun (sing. or mass)
Adverb Adjective Noun (sing. or mass)
Proper noun (sing.) Proper noun (sing.) Noun (sing. or mass)
Adjective Noun (sing. or mass) Verb (gerund/present participle)
Adjective Adjective Adjective Noun (pl.)
Adjective Adjective Adjective Noun (sing. or mass)
Noun (sing. or mass) Adjective Noun (pl.)
Noun (pl.) Noun (sing. or mass)
Proper noun (sing.) Verb (3rd pers. sing. present)
Proper noun (sing.) Adjective Noun (sing. or mass)
Adjective Proper noun (sing.)
Cardinal number Noun (sing. or mass)
Symbol
Proper noun (sing.) Noun (sing. or mass) Noun (sing. or mass) Noun (sing. or mass)
Adjective Noun (sing. or mass) Verb (gerund/present participle) Noun (sing. or mass)
Adjective Adjective

Figure 4.1: *PoS patterns of manually assigned keywords present in abstract, where the pattern occurs ten or more times in the training data.*

4.2.2 DEFINING THE FEATURES

The investigated features for the candidate terms are the three features proposed by Frank *et al.* (1999), that is, term frequency (TF), inverse document frequency (IDF), and the position of the first occurrence, where the position is normalised by the length of the document. Whereas Frank *et al.* combine the TF and the IDF values, these are in this investigation kept as two distinct features.

The inverse document frequency (Sparck Jones, 1972) is calculated with the following formula:

$$idf_i = \log\left(\frac{N}{n_i}\right)$$

where N denotes the number of documents in the collection, and n_i is the number of documents in which term i occurs.

In addition, a fourth feature is investigated, namely the PoS tag or tags assigned to a candidate term. In case the candidate term consists of several tokens, the PoS tags are treated like a sequence. As an example, an extracted phrase like *random*_JJ *excitations*_NNS gets the atomic feature value JJ_NNS. In case a term occurs more than once in a document, the tag or tag sequence assigned is the most frequently occurring one for that term in the entire document. In case of a draw, the first occurring one is assigned.

4.3 THE DATA REPRESENTED

When the feature values have been calculated for each extracted candidate term, the calculated values are put in a vector (one vector per candidate term). For the learning, only the feature values are considered, and the term itself has only the function of being an identifier. In the machine learning field, a feature value vector is called *an example*. For the experiments described in this chapter, examples that are manually assigned keywords are assigned the class *positive*, while all other terms are *negative* examples.

This section will describe an investigation of the suggested representations. For each term selection approach, the number of manually assigned keywords present in the documents that is retained is studied, to know the maximum number of manually assigned keywords that can be found by the learning algorithm. Also, the distribution of the feature values for the four features is discussed for each one of the term selection approaches.

4.3.1 FINDING THE KEYWORDS

Table 4.1 summarises how good each term selection approach is at finding the manually assigned keywords that are present in the documents. As mentioned, not all of the manually assigned keywords are present in the abstract to which the keyword is assigned. It is the number of keywords present that is used for calculating the recall presented in Table 4.1. In other words, 100% recall is possible even if there are manually assigned keywords that are not present in the abstract. The recall values in Table 4.1 define the upper-bound for the recall of the subsequent machine learning. The numbers in the table are those for the stemmed candidate terms. If the term selection approaches were not the first step in representing the data for a subsequent machine learning algorithm, but instead were methods for selecting appropriate keywords, this is the performance that would be achieved.

Table 4.1: *Results if all extracted candidate terms for each approach were to be assigned to the documents. In the table is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold.*

Approach	**Assign. tot.**	**Assign. mean**	**Corr. tot.**	**Corr. mean**	**P**	**R**	**F**
n-grams	93 919	93.9	6 514	6.5	6.9	86.2	12.8
NP-chunks	31 946	31.9	3 771	3.8	**11.8**	49.9	**19.1**
PoS patterns	69 973	70.0	6 664	6.7	9.5	**88.2**	17.2

The n-gram approach extracts in total 93 919 stemmed candidate terms, while the NP-chunk approach extracts 32 138 unstemmed and 31 946 stemmed terms, and the PoS pattern approach extracts 73 316 unstemmed and 69 973 stemmed terms. The largest amount of keywords remains when extracting terms matching the PoS patterns (88.2%). When extracting NP-chunks half of the keywords are lost. The n-gram approach keeps 86.2%. When applying stemming, the proportion of positive examples increases slightly.

In Table 4.1, the proportion of the extracted candidate terms being manually assigned keywords is also shown (precision). As can be noted, the amount of incorrect terms (that is, not manually assigned keywords) that is extracted by each method is very high. The largest proportion of negative examples is obtained by the n-gram approach.

For the example sentence in the previous section, the highest number of candidate terms is extracted by the PoS pattern approach. This is, however, not the case for the data set investigated, where the n-gram approach extracts the largest number of candidate terms.

4.3.2 DISTRIBUTION OF THE FEATURE VALUES

This section will investigate the distribution of the feature values for the four features for each term selection approach. This will be done by displaying the proportion of the positive and the negative examples respectively having a specific feature value, as well as by describing the results of the statistical tests that have been performed. For the statistical tests, the absolute numbers of examples having a certain feature value are used, and not the proportion. As the distribution looks rather similar for the unstemmed and the stemmed candidate terms, only the feature value distributions of the stemmed terms are described.

Term frequency

Figures 4.2–4.4 show the term frequency for the extracted n-grams, NP-chunks, and the terms matching any of the PoS patterns respectively. In

these figures, the proportions of the negative and the positive examples having a certain frequency are shown. In the figures, only TF$\leq$5 are included, as the higher term frequencies are not visible anyway.

For the n-gram approach and the NP-chunk approach, the term frequency differs mainly for frequency 1 and 2, where a larger proportion of the positive terms occurs twice in a document. As can be seen in Figure 4.2 and Figure 4.3, the distribution is also skewed for the higher term frequencies. The difference in distribution of the term frequency values for the positive and the negative examples is statistically significant for the n-gram approach and the NP-chunk approach according to a t-test ($p<0.0001$). As for the PoS pattern approach, the proportion of positive and negative examples having a certain term frequency value is rather similar for the two classes (see Figure 4.4). This difference is not statistically significant according to a t-test. It should also be noted that as much as 80% of the manual keywords present in its abstract occur only once.

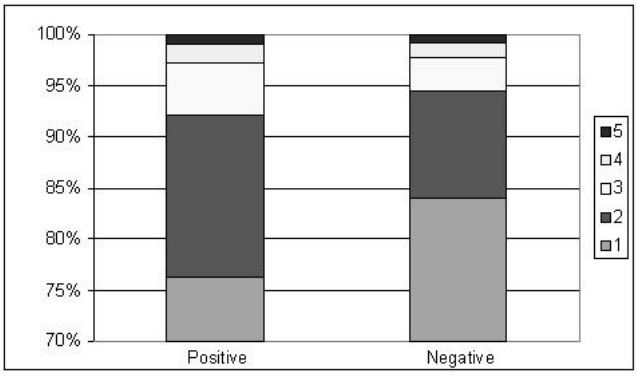

Figure 4.2: *Term frequency for the* n*-gram approach. The* y*-axis denotes the proportion of the positive and the negative examples having a term frequency of 1, 2, 3, 4, or 5.*

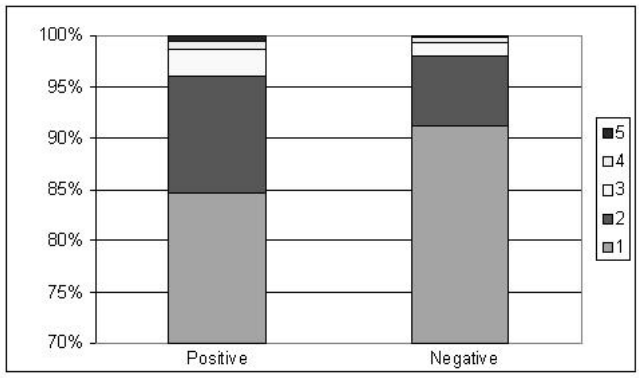

Figure 4.3: *Term frequency for the NP-chunk approach. The* y*-axis denotes the proportion of the positive and the negative examples having a term frequency of 1,2, 3, 4, or 5.*

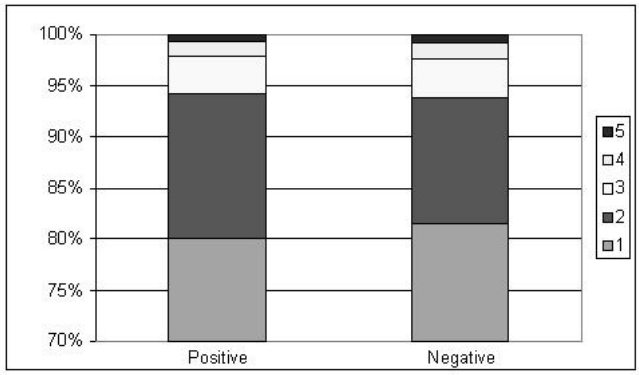

Figure 4.4: *Term frequency for the PoS pattern approach. The* y*-axis denotes the proportion of the positive and the negative examples having a term frequency of 1, 2, 3, 4, or 5.*

Inverse document frequency

Figures 4.5–4.7 display the IDF values for the three term selection approaches respectively. In the tables, the x-axis denotes the IDF value, and the y-axis denotes the proportion of the positive and the negative examples having a certain IDF value. When the IDF equals 3, a candidate term occurs in only one document in the collection. For the three approaches, the curves look rather similar. Also, the distribution for the negative and the positive examples is rather similar up to around 2.5. There are clearly fewer negative examples in the higher ranges for all three term selection approaches. The positive examples are, in other words, more unique in the collection. The difference in distribution of the IDF values for the positive and the negative examples is statistically significant for all three term selection approaches according to a t-test ($p<0.0001$).

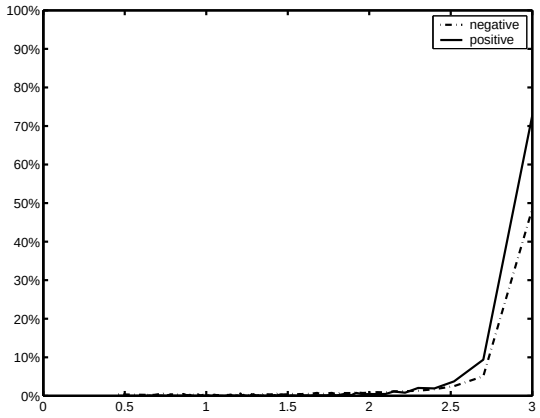

Figure 4.5: *Inverse document frequency for the* n*-gram approach. The* x*-axis denotes the IDF value. The* y*-axis denotes the proportion of the positive and the negative examples having a certain IDF value.*

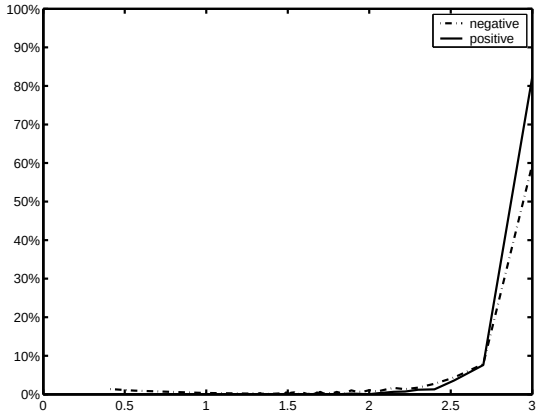

Figure 4.6: *Inverse document frequency for the NP-chunk approach. The* x*-axis denotes the IDF value. The* y*-axis denotes the proportion of the positive and the negative examples having a certain IDF value.*

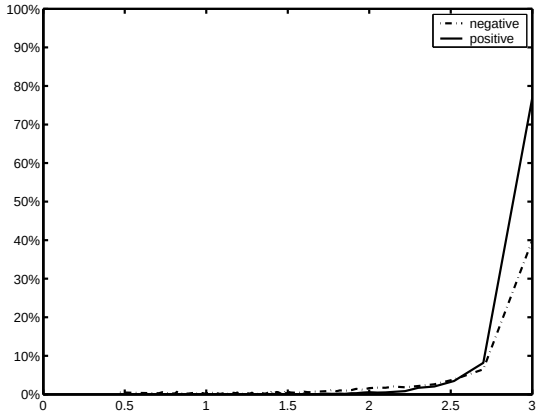

Figure 4.7: *Inverse document frequency for the PoS pattern approach. The* x*-axis denotes the IDF value. The* y*-axis denotes the proportion of the positive and the negative examples having a certain IDF value.*

Position of first occurrence

The relative position of the first occurrence of a candidate term has a more even distribution throughout a document for the negative examples compared to the positive examples, where the positive examples occur more often in the beginning of a document. This is shown in Figures 4.8–4.10, where the x-axis denotes the proportion of a document preceding the first occurrence. The difference in distribution of the feature values for the relative first occurrence for the positive and negative candidate terms is statistically significant according to a t-test ($p<0.0001$).

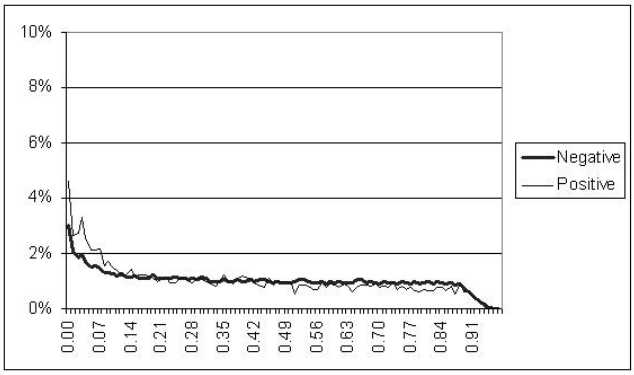

Figure 4.8: *Relative first position for the* n*-gram approach. The* x*-axis denotes the proportion of a document preceding the first occurrence. The* y*-axis denotes the proportion of the positive and the negative examples occurring at a certain position.*

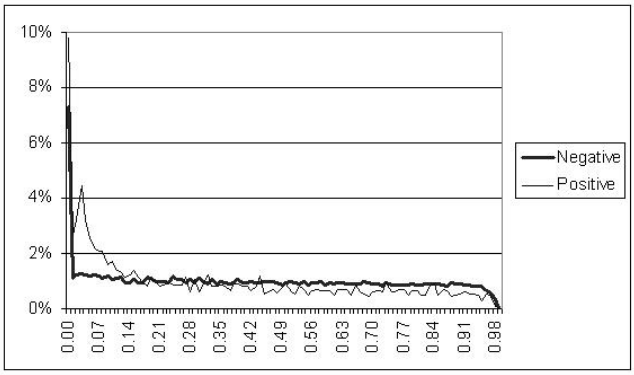

Figure 4.9: *Relative first position for the NP-chunk approach. The* x*-axis denotes the proportion of a document preceding the first occurrence. The* y*-axis denotes the proportion of the positive and the negative examples occurring at a certain position.*

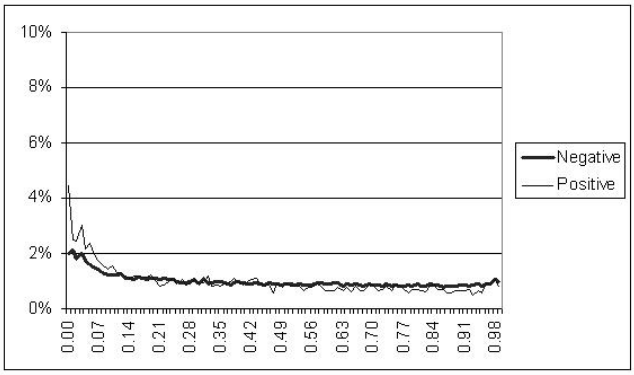

Figure 4.10: *Relative first position for the PoS pattern approach. The* x*-axis denotes the proportion of a document preceding the first occurrence. The* y*-axis denotes the proportion of the positive and the negative examples occurring at a certain position.*

PoS tags

Tables 4.2–4.4 show the five most frequently occurring PoS tags for the candidate terms that are negative and positive examples respectively. The number of values (that is, different PoS tag or tags) that this feature takes varies for the three approaches: the n-gram approach has 1 498 different values, the NP-chunk approach has 858 different values, while the PoS pattern approach has 116 different values. The reason why the PoS pattern approach has more than 56 different values (which is the number of defined patterns) is that once a candidate term has been extracted, the most frequently occurring tag(s) for that candidate term in the entire document is assigned.

As mentioned in Section 4.2.1, the PoS tags follow the Penn tree bank convention[2]. From these tables, it is clear that certain PoS patterns are more likely to denote a positive example than a negative example, and vice versa. To give an extreme example, the most frequently occurring pattern for the NP-chunks is 'determiner noun' (about 17%), but this PoS tag sequence is not a keyword at any occasion in the examined data set. According to a χ^2 test, the difference in distribution for the feature values over the positive and the negative examples for each term selection approach is statistically significant ($p<0.005$).

Table 4.2: *The distribution of the five most frequently occurring PoS tags for the negative (Neg.) and the positive (Pos.) examples respectively for the* n*-gram approach.*

Tag(s)	**Neg. (%)**	**Pos. (%)**	**Tag(s)**	**Pos. (%)**	**Neg. (%)**
NN	18.8	7.5	JJ_NN	15.4	6.0
JJ	10.3	1.2	NN_NN	13.5	3.5
NNS	6.7	3.4	JJ_NNS	10.3	1.8
JJ_NN	6.0	15.4	NN	7.5	18.8
NN_NN	3.5	13.5	NN_NNS	7.0	1.6

[2]Thus, in Tables 4.2–4.4, NN stands for a singular or a mass noun, NNS is a plural noun, JJ is an adjective, and finally, DT stands for determiner.

Table 4.3: *The distribution of the five most frequently occurring PoS tags for the negative (Neg.) and the positive (Pos.) examples respectively for the NP-chunk approach.*

Tag(s)	**Neg. (%)**	**Pos. (%)**	**Tag(s)**	**Pos. (%)**	**Neg. (%)**
DT_NN	16.8	0.0	JJ_NNS	14.5	4.6
NN	9.6	7.4	NN_NN	11.4	1.7
DT_JJ_NN	8.5	0.0	JJ_NN	11.4	3.2
NNS	6.7	5.0	NN_NNS	9.8	2.0
JJ_NNS	4.6	14.5	NN	7.4	9.6

Table 4.4: *The distribution of the five most frequently occurring PoS tags for the negative (Neg.) and the positive (Pos.) examples respectively for the PoS pattern approach.*

Tag(s)	**Neg. (%)**	**Pos. (%)**	**Tag(s)**	**Pos. (%)**	**Neg. (%)**
NN	26.6	7.4	JJ_NN	15.1	8.4
JJ	16.9	0.6	NN_NN	13.9	5.4
NNS	10.1	3.4	JJ_NNS	10.8	3.3
JJ_NN	8.4	15.1	NN_NNS	7.4	2.4
NN_NN	5.4	13.9	NN	7.4	26.6

4.4 DISCUSSION

This chapter has presented an investigation of different representations to be used as input to an algorithm that will learn to automatically extract keywords. In more detail, three approaches for selecting candidate terms from written documents, as well as the distribution of the values for four features have been examined. The three term selection approaches are n-grams, NP-chunks, and PoS patterns. The four explored features are term frequency, inverse document frequency, relative position of first occurrence, and the PoS tag or tags assigned to the candidate term.

The pre-processed data will subsequently be the input to a supervised machine learning algorithm. The training and the evaluation of the learning

is the focus of the next chapter. The recall values for the term selection approaches reported in this chapter (in Section 4.3.1) denote the upper-bound for the learning algorithm when it comes to the recall of the selected keywords. The purpose of the learning is to increase the precision — which is very low — while still retaining as many correctly extracted keywords as possible (that is, retaining a high recall).

Compared to Turney (2000) and Frank *et al.* (1999) who experiment with keyword extraction to and from full-length texts, the data set investigated in this dissertation differs in one important aspect, namely that it consists of abstracts only. This means that the texts are shorter, and also that they are denser in information. From the point of view of learning to automatically extract keywords, the task can be both easier and harder: it might be easier in the sense that there will be less noise, that is, fewer incorrect candidate terms will be extracted from the documents. It might, however, also be harder, as the features — and more specifically, the term frequency and the relative first position — will convey less reliable information. In a longer document, the term frequency for keywords might be higher than for non-keywords, and as the documents are longer, a keyword might be more likely to appear earlier in the document.

Chapter 5

Classification

In this chapter, an evaluation of a number of prediction models for automatic keyword extraction is described. By adding linguistic knowledge to the representation (such as syntactic features), rather than relying only on statistics (such as term frequency), better performance can be obtained as measured by keywords previously assigned by professional indexers.

The experiments discussed in this chapter have previously been described in the paper *Improved automatic keyword extraction given more linguistic knowledge* presented at the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP) held in Sapporo, Japan (Hulth, 2003b) (parts of which were also presented in Chapter 4).

5.1 Introduction

As could be seen in Chapter 4, the number of candidate terms extracted by the three term selection approaches is large. More importantly, the precision is low — as many of the candidate terms are not manually assigned keywords — ranging from 6.9 for the n-gram approach to 11.8 for the NP-chunk approach. To reduce the number of candidate terms, a machine learning algorithm is applied. The goal for the learning is to find what

feature values discriminate candidate terms that are keywords from those that are not. These experiments will now be described.

5.1.1 TERM SELECTION APPROACHES

As described in Chapter 4, three term selection approaches have been defined. These are n-grams; NP-chunks; and terms matching any of a set of part-of-speech tag sequences.

The n-gram approach

In the first term selection approach, the terms are defined in a manner similar to Turney (2000) and Frank *et al.* (1999). (Their studies are introduced in Section 2.2.) All unigrams, bigrams, and trigrams are extracted. Thereafter a stop list is applied (from Fox (1992)), and all terms beginning or ending with a stop word are removed. Finally all remaining tokens are stemmed using Porter's stemmer (Porter, 1980).

The NP-chunk approach

For the second term selection approach a chunker is used to select all NP-chunks from the documents. Experiments with both unstemmed and stemmed terms are performed.

The PoS pattern approach

For the third term selection approach, a set of PoS tag patterns — in total 56 — are defined, and all (part-of-speech tagged) words or sequences of words matching any of these are extracted. The patterns are those PoS tag sequences of the manually assigned keywords — present in the training data — occurring ten or more times. As with the NP-chunk approach, experiments with both unstemmed and stemmed terms are performed.

5.1.2 FEATURES

For a number of runs, the same features that Frank *et al.* (1999) use for their domain-independent experiments are used. These are:

- term frequency
- inverse document frequency
- relative position of the first occurrence

The representation differs in that the term frequency and the IDF are not weighted together, but kept as two distinct features. In addition, the real values are not discretised, only rounded off to two decimals, thus more decision-making is handed over to the learning algorithm. The IDF is calculated for the three data sets separately.

In addition, experiments with a fourth feature are performed. This is the PoS tag or tags assigned to the candidate term by the same chunker used for finding the chunks and the tag patterns. When a term consists of several tokens, the tags are treated like a sequence. In case a term occurs more than once in a document, the tag or tag sequence assigned is the most frequently occurring one for that term in the entire document. In case of a draw, the first occurring one is assigned.

5.2 TRAINING AND EVALUATION

The feature values are calculated for each extracted candidate term in the training and the validation sets, that is for the n-grams, the NP-chunks, the stemmed NP-chunks, the terms matching the PoS patterns, and the stemmed terms matching the PoS patterns respectively. In other words, the term frequency, the inverse document frequency, and the proportion of the document preceding the first appearance for each candidate term are calculated. Also, the PoS tag(s) for each candidate term are extracted. In addition, as the machine learning approach is supervised, each candidate term is assigned its class, that is, information on whether the candidate

term is a manually assigned keyword or not is added. For the stemmed terms, a candidate term is considered a keyword if it is equivalent to a stemmed manually assigned keyword. For the unstemmed terms, the term has to match exactly.

In these experiments, the main concern is the precision and the recall for the examples that have been assigned the class *positive*, that is the amount of the suggested keywords that is correct (precision), and the amount of the manually assigned keywords that is found (recall). (See Section 3.4 for a definition of these two measures.) When calculating the recall, the value for the total number of manually assigned keywords present in the documents is used, independently of the number actually present in the different representations. This figure varies slightly for the unstemmed and the stemmed data, and for the two, the corresponding value is used.

Several runs are made for each representation, with the goal to maximise the performance as evaluated on the validation set. First the weights of the positive examples are adjusted, as the data set is unbalanced. Better performance is obtained when the positive examples in the training data outnumber the negative ones. Thereafter experiments with bagging are performed, and also, runs with and without the PoS tag feature are made. The results for the experiments are presented next.

5.3 RESULTS

In this section, the results obtained on the previously unseen test set, by the best performing model for each representation — as judged on the validation set — are presented. It should, however, be noted that the number of possible runs is very large, by varying for example the number of classifiers generated by the ensemble technique. It might well be that the optimal setting has not been found for any of the representations.

The length of the abstracts in the test set varies from 23 to 338 tokens (the median is 121 tokens). The number of uncontrolled keywords per document is 2 to 31 (the median is 9 keywords). The total number of stemmed manual keywords present in the stemmed test set is 3 816, and the average

number of keywords is 7.6. Their distribution over the 500 documents is 0 (for 3 documents) to 27, and the median is 7.

As for bagging, it has been noted in these experiments that although the accuracy (that is, the number of correctly classified positive and negative examples divided by the total number of examples) improves when increasing the number of classifiers, the F-measure often decreases. For the PoS pattern approach without the tag feature the best model consists of a 5-bagged classifier, for the PoS pattern approach with the tag feature a 20-bagged, and finally for the n-gram approach with the tag feature a 10-bagged classifier. For the other three runs a single classifier has the best performance.

Since stemming with few exceptions leads to better results on the validation set for all runs, only these values are presented in this section. In Table 5.1, the number of assigned terms and the number of correct terms, in total and on average per document are shown. Also, precision, recall, and the F-measure are presented. For each approach, both the results with and without the PoS tag feature are given.

Table 5.1: *For each representation is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F). The highest values are shown in bold. The total number of manually assigned terms present in the abstracts is 3 816, and the mean is 7.6 terms per document.*

Approach	**Assign. tot.**	**Assign. mean**	**Corr. tot.**	**Corr. mean**	**P**	**R**	**F**
n-grams	21 104	42.2	2 187	4.4	10.4	57.3	17.6
n-grams w. tag	7 815	15.6	1 973	3.9	25.2	51.7	**33.9**
NP-chunks	8 189	16.4	1 364	2.7	16.7	35.7	22.7
NP-chunks w. tag	4 788	9.6	1 421	2.8	**29.7**	37.2	33.0
PoS patterns	15 882	31.8	2 519	5.0	15.9	**66.0**	25.6
PoS patterns w. tag	7 012	14.0	1 523	3.0	21.7	39.9	28.1

5.3.1 *n*-GRAMS

When extracting terms from the test set according to the *n*-gram approach, the data consist of 42 159 negative examples, and 3 330 positive examples, thus in total 45 489 examples are classified by the trained model. Using this manner of extracting the terms means that 12.7% of the manual keywords originally present in the test set are lost.

To summarise the *n*-gram approach (see Table 5.1), without the tag feature it finds on average 4.4 keywords per document, out of originally on average 7.6 manual keywords present in the abstracts. However, the price paid for these correct keywords is high: almost 38 incorrect per document. When adding the PoS tag feature, the number of correct keywords decreases slightly, while the number of incorrect keywords decreases to a third. If looking at the actual distribution of assigned keywords for these two runs, this varies between 5 and 134(!) without the tag feature, and from 1 to 48 with the tag feature. The median is 40 and 14 respectively. The F-measures for these two runs are 17.6 and 33.9 respectively. 33.9 is the highest F-measure that is achieved for the six runs presented here.

5.3.2 NP-CHUNKS

When extracting the terms according to the stemmed NP-chunk approach, the test set consists of 13 579 negative, and 1 920 positive examples; in total 15 499 examples.

An F-measure of 22.7 is obtained without the PoS tag feature, and 33.0 with this feature. The number of keywords on average per document is 16.4 without the tag feature, and 9.6 with it. If looking at each document, the number of keywords assigned varies from 0 (for 3 documents) to 46 with a median of 16, and 0 (for 4 documents) to 29 with the median value being 9 keywords.

Extracting the terms with the NP-chunk approach means that slightly more than half of the keywords actually present in the test set are lost, and compared to the *n*-gram approach the number of correct terms assigned is almost halved. The number of incorrect keywords, however, decreases

considerably. But, the difference in performance is shown when the PoS tag feature is included: the number of correctly assigned terms is more or less the same for this approach with or without the tag feature, while the number of incorrect terms is halved.

5.3.3 PoS patterns

When extracting the stemmed terms according to the PoS pattern approach, the test data consist of 33 507 examples. Of these are 3 340 positive, and 30 167 negative. In total, 12.5% of the keywords present are lost.

The F-measures for the two runs, displayed in Table 5.1, are 25.6 (without the tag feature) and 28.1 (with the tag feature). The number of keywords assigned on average per document is 5.0 and 3.0 without and with the tag feature respectively. The actual number of keywords assigned per document is 0 (for 3 documents) to 100 without the tag feature, and 0 (for 4 documents) to 46 with the tag feature. The median is 30 and 12 respectively.

5.4 Discussion

This chapter has shown how keyword extraction from abstracts can be achieved by using statistical measures as well as syntactic information from the documents, as input to a machine learning algorithm. If first considering the term selection approaches, the highest F-measure is obtained by one of the n-gram runs. One of the NP-chunk runs results in the highest precision, while one of the PoS pattern approaches gives the highest recall, where the largest amount of manually assigned keywords present in the abstracts is selected by the PoS pattern approach without the tag feature. The PoS pattern approach is also the approach that retains the largest number of assigned terms after that the data have been pre-processed. Using phrases means that the length of the candidate terms is not restricted to something arbitrary; rather the terms are treated as the units they are. However, of the PoS patterns defined for the experiments discussed in this dissertation none is longer than four tokens. If looking at all stemmed manually assigned

keywords present in the test set, 2.2% are then ruled out as candidate terms. The longest NP-chunks from the test set that are correctly assigned are five tokens long. As for when syntactic information is included as a feature (in the form of the PoS tag(s) assigned to the term), it is evident from the results presented in this chapter that this information is crucial for assigning an acceptable number of terms per document, independently of which term selection approach that is chosen.

One shortcoming of the work is that there is no relation between the different PoS tag feature values. For example, a singular noun has no closer relationship to a plural noun than to an adjective. This could be solved by removing morphological information, or by categorising the patterns in a way that reflects their semantics, perhaps in a hierarchical manner.

In the next chapter, experiments on how the number of incorrectly selected keywords is reduced (increasing the precision), while not loosing too many of the correctly selected keywords (retaining the recall) are presented.

Chapter 6

Expert Combination

This chapter extends the experiments on automatic keyword extraction described in Chapter 5, by showing how the number of incorrectly assigned keywords may be highly reduced, as measured by keywords assigned by professional indexers. This is achieved by combining the predictions of several classifiers. The classifiers are trained on different representations of the data, where the difference lies in the definition on what constitutes a candidate term in a written document.

The experiments on expert combination described in this chapter have previously been presented in the book chapter *Reducing false positives by expert combination in automatic keyword indexing*, published in *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003* (Hulth, 2004a).

6.1 Introduction

The most evident drawback with the classifiers evaluated in Chapter 5 is that they select too many keywords that are not chosen as keywords by the human indexers, that is, the precision is too low. In some cases it may be desirable to have a prediction model that assigns more keywords

than a human would do, for example if the classifier is to be used for semi-automatic indexing. In that case the goal for the training should be to find all manually assigned keywords, as well as a limited set of additional ones. The final selection would then be made by a professional indexer. Semi-automatic indexing is, however, not the purpose of the experiments described in this dissertation.

In this chapter, experiments on how the number of incorrectly assigned keywords is reduced to a more acceptable level are described. The improvement is obtained by taking the majority vote of three classifiers which each is trained on a different representation of the data. The representations differ in how the candidate terms are selected from the documents; using the different definitions on what constitutes a term in a written text.

In the experiments presented in Chapter 5, in total six models are evaluated on the test set (three term selection approaches with three and four features respectively). The results from this evaluation that are relevant for the experiments discussed in this chapter are found in Table 6.1. The presented results are for models based on four features, that is, the term frequency, the inverse document frequency, the relative position of the first occurrence, and the PoS tag or tags assigned. The total number of manually assigned keywords present in the abstract is 3 816, and the mean is 7.6 keywords per document. Of these classifiers are two bagged: the PoS pattern approach consists of a 20-bagged model, and the n-gram approach consists of a 10-bagged model.

6.2 ENSEMBLE METHODS

It has often been shown in machine learning that combining prediction models leads to an improved accuracy, and there are several ways to apply ensemble techniques (see for example Dietterich (1998)). Basically, one may either manipulate the training data; for example in both *bagging* and *boosting* a number of classifiers are obtained by training on different subsets of the whole data set. Or, one may use different learning algorithms on the same training data to acquire different classifiers. There is also the question of how to combine the classifiers to consider, for example whether better

Table 6.1: *Results for the individual classifiers. For each term selection approach is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold.*

Approach	Assign. tot.	Assign. mean.	Corr. tot.	Corr. mean.	P	R	F
n-grams	7 815	15.6	1 973	3.9	25.2	**51.7**	**33.9**
NP-chunks	4 788	9.6	1 421	2.8	**29.7**	37.2	33.0
PoS patterns	7 012	14.0	1 523	3.0	21.7	39.9	28.1

performing classifiers should be given higher weights in the ensemble. (For a review on the latter, see for example Bahler and Navarro (2000).)

Experiments on how different retrieval results can be automatically combined have been presented by Strzalkowski *et al.* (1999). The goal of those experiments was to improve traditional information retrieval by using several indexing methods to retrieve different sets of documents, and then to merge (or combine) these results into one single result. The merging method applied is an implementation of an algorithm called *Sequel* (Asker and Maclin, 1997). The rationale behind Sequel is to find the most confident classifier down to a certain threshold. It requires that the list is sorted by — in this case — relevance score. The confidence is calculated by finding the classifier with the highest proportion of correct classifications (that is, relevant documents) at the top, down to the first non-relevant one. The threshold is the lowest relevance score within this interval. The items covered by the span are then removed from all classifiers.

6.3 COMBINING DIFFERENT REPRESENTATIONS

In this section, an ensemble method that highly reduces the number of incorrectly assigned keywords, while still retaining a large number of correct keywords, will be described. The ensemble is built from classifiers trained

on different representations of the data, and more specifically on different definitions on what constitutes a candidate term in a written document.

The classifiers used in the experiments described in this chapter are trained and evaluated on the same data as in the previous experiments (described in Chapter 5): a set of 2 000 abstracts in English, from scientific journal papers with keywords assigned by professional indexers. Also, the same division of the training (1 000 documents), validation (500 documents), and test (500 documents) sets is kept.

As mentioned in Section 6.1, six models are evaluated on the test set in the previous chapter: three term selection approaches with two sets of features (with or without the PoS tag feature). To reduce the number of incorrect keywords a pair-wise combination is made over the six models, provided that the term selection approaches are different. Thus, in total twelve pairs are obtained. In order to be considered a keyword, a term has to be selected by both prediction models in the pair, that is, an unanimity vote is used.

The twelve pairs are evaluated on the validation data, and the three pairs (one pair for each combination of the term selection approaches) with the highest precision are selected. The reason for choosing precision as the selection criteria is that the goal is to reduce the false positives[1], thus the proportion of these should be as small as possible.

The three selected pairs are:

- n-grams with the PoS tag feature + NP-chunks with the PoS tag feature
- n-grams with the PoS tag feature + PoS patterns with the PoS tag feature
- NP-chunks with the PoS tag feature + PoS patterns with the PoS tag feature

In other words, the three term selection approaches with the PoS tag feature. In Table 6.2, the results for these three pair-wise combinations on the test

[1] A *false positive* is a negative example that has wrongly been given the class positive.

Table 6.2: *Results for the three best performing pairs. In the table is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold.*

Pair	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
n-gram+NP-chunk	2 004	4.0	902	1.8	**45.0**	23.6	**31.0**
n-gram+PoS pattern	3 822	7.6	1 069	2.1	28.0	**28.0**	28.0
NP-chunk+PoS pattern	1 684	3.4	708	1.4	42.0	18.6	25.7

set are shown. In the table, the number of assigned keywords in total; the number of correct keywords in total; the precision; the recall; and the F-measure are displayed. The total number of manually assigned keywords present in abstract in the test data is 3 816, and the mean is 7.6 keywords per document. As can be seen in Table 6.2, n-grams + NP-chunks assign the set of 500 abstracts in total 2 004 keywords, and on average 4.0 keywords per document. Of these are on average 1.8 correct. If looking at the actual number of keywords assigned, 27 documents have 0 keywords, while the highest number of keywords assigned is 21. The median is 4. For the n-grams + PoS patterns pair, in total 3 822 keywords are assigned. Of the 7.6 keywords on average per document, 2.1 are correct. If examining the actual distribution, 8 documents have no keywords assigned, the maximum is 34, and the median is 7. Finally, the NP-chunks + PoS patterns assign in total 1 684 keywords, and on average 3.4 keywords per document; of these are 1.4 correct. A maximum of 14 keywords is actually assigned. 32 documents have no selected keywords, and the median is 3 keywords per document.

As can be seen in Table 6.2, the precision increases for all pairs, compared to the performance of the individual classifiers (see Table 6.1). However, the F-measure decreases for all three combinations. As it is important not only to assign correct keywords, but also to actually find the manually assigned keywords, recall is considered equally important (the reason for when calculating the F-measure giving β the value 1).

As these results are still not satisfactory from the point of view of the F-measure, an experiment with joining the results of the three pairs has been performed, that is, by taking the union of the keywords assigned by the three pairs. Doing this is in fact equivalent to taking the majority vote of the three individual classifiers in the first place. The results on the test set when applying the majority vote to the three individual classifiers are found in Table 6.3 (Subsumed not removed). In total, 5 352 keywords are then assigned to the test set. On average per document, 3.3 keywords are correct, and 7.4 are incorrect. The actual number of keywords varies between 0 (for 4 documents) and 41, and the median is 10 keywords per document. The F-measure when taking the majority vote for the three classifiers is 36.1, thus higher than for any of the individual classifiers. The precision is also higher than for any of the component classifiers, and the recall is higher than for two of the individual classifiers. If comparing this result to the *n*-gram approach, that has the highest F-measure of the individual classifiers, the number of false positives decreases by 2 145, while 318 true positives are lost. The performance of the majority vote may be compared to the number of keywords assigned to the test set by the professional indexers, where 3 documents have no keywords present in its abstract. The median is 7 keywords, and the maximum is 27, and there are in total 3 816 manually assigned keywords.

Table 6.3: *Results for the majority vote. In the table is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold. In the table,* Sub. *stands for* Subsumed.

Majority vote	**Assign. tot.**	**Assign. mean**	**Corr. tot.**	**Corr. mean**	**P**	**R**	**F**
Sub. not removed	5 352	10.7	1 655	3.3	30.9	**43.4**	36.1
Sub. removed	4 369	8.7	1 558	3.1	**35.7**	40.8	**38.1**

As another improvement, the subsumed keywords may be removed, that is, if a term is a substring of another assigned keyword, the substring is removed. (This is also done in the experiments performed by

Frank *et al.* (1999).) In Table 6.3 (Subsumed removed) one can see that although some correctly assigned keywords are removed as well (5.9%), the number of false positives decreases by 24.0%. If looking at the actual distribution on the test set, 4 documents have 0 keywords. A maximum of 30 keywords is assigned, while the median is 8 keywords per document. This results in the highest F-measure (38.1) obtained on the test set for the experiments on expert combination described in this chapter.

It is interesting to investigate how well the term selection approaches perform alone, that is, when not applying any machine learning. In other words, to extract all terms according to the three approaches, and then take the majority vote and remove the subsumed terms. These results are presented in Table 6.4. It is noteworthy that by combining the three term selection approaches alone, a higher F-measure (34.2) is achieved than for any of the individual models — where the n-gram approach has an F-measure of 33.9 (see Table 6.1) — provided that the subsumed keywords are removed! (Note, however, that for the individual classifiers, subsumed terms are not removed.)

Table 6.4: *Results when assigning all candidate terms selected by at least two term selection approaches, without any prediction models. In the table is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold.*

No prediction	**Assign. tot.**	**Assign. mean**	**Corr. tot.**	**Corr. mean**	**P**	**R**	**F**
Sub. not removed	30 684	61.4	3 346	6.7	10.9	**87.7**	19.4
Sub. removed	9 343	18.7	2 251	4.5	**24.1**	59.0	**34.2**

6.4 DISCUSSION

This chapter has shown how the number of automatically assigned keywords that are incorrect can be highly reduced, while the number of correct keywords is still satisfactory, as measured by keywords assigned by professional indexers. The improvement is achieved by taking the majority vote for three classifiers, each trained on a different representation of the data. The representations differ in how the candidate terms are selected from the documents. The three methods extract uni-, bi-, and trigrams; NP-chunks; and terms matching any of a set of PoS patterns (see Chapter 4 for details). If precision for some reason is considered more important than the F-measure — if the set of assigned keywords must have a high quality — a combination of either n-grams and NP-chunks, or NP-chunks and PoS patterns, using an unanimity vote should be used instead.

Before any successful results were obtained on the task of reducing the number of incorrectly assigned keywords, two other methods were examined. In these two experiments, combinations of classifiers were made for each term selection approach separately. In the first unsuccessful experiment, *bagging* (Breiman, 1996) was applied, that is, from a training set consisting of n examples, a new set of the same size is constructed by randomly drawing n examples with replacement. This procedure is repeated m times to create m classifiers. Both voting, with varying numbers of classifiers that should agree, as well as setting varying threshold values for the number of maximum keywords to assign to each document in combination with voting, was tried.

In the second unsuccessful experiment, the fact that the data set is unbalanced was exploited. By varying the weights given to the positive examples for each run, a set of classifiers was obtained. Thereafter voting was applied in the same fashion as for the first unsuccessful experiment. In addition, a simple weighting scheme was applied, where higher weights were given to classifiers that found more correct keywords.

As the results for these experiments were poor (on the validation data), they are not presented here. Although the number of false positives did decrease, too many of the true positives also disappeared. As these experiments did not succeed, it may be suspected that the selected features are

not enough to discriminate keywords from non-keywords, at least not in the data set used for these experiments.

One precondition for an ensemble performing better than the individual prediction models is that the errors made by the different classifier are uncorrelated (Dietterich, 1998). To establish which errors that are specific for one term selection approach, while not present in the two other (which types of errors that are avoided by taking the majority vote), all false positives from ten arbitrarily selected documents in the validation set have been collected. It is, however, difficult to categorise the errors made by each approach. One of the few things that can be noted is that some of the terms selected by the NP-chunk approach begin with a determiner (for example 'a given grammar'); extracted per definition. However, these types of terms are rarely keywords, and are not extracted by the two other approaches (most determiners are stop words, and are not part of the PoS patterns).

An alternative to taking the majority vote could be to rank the keywords according to how many of the individual classifiers that agree upon a candidate term being a keyword, thus obtaining a probabilistic output. First the classifiers would need to be ranked internally according to their previous performance. A threshold value for the maximum number of keywords to assign to each document would then have to be set, either by the system or by a user.

When inspecting the automatically assigned keywords, it may be concluded that using keywords assigned by one professional indexer as the gold standard for the evaluation does not always give justification to the models. Many automatically selected keywords have a clear relation to the subject at hand, although they were not chosen by the human for one reason or another.

Chapter 7

Regression

In this chapter, three minor alterations to the keyword extraction algorithm are suggested, and the results of the empirical verifications are presented. Also, one major change is made. By treating the task as a regression task instead of a classification task, better performance is achieved, as measured by keywords previously assigned by professional indexers.

The experiments presented in this chapter are also discussed in *Enhancing linguistically oriented automatic keyword extraction* published in the Proceedings of the Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Annual Meeting (HLT/NAACL), held in May in Boston, USA, and presented as a poster at this conference. (Hulth, 2004b).

7.1 Introduction

In the next section of the chapter, three minor modifications made to the keyword extraction algorithm will be presented. The main focus of this chapter is, however, how the learning task is defined. Instead of making a binary classification, where a candidate term is either a keyword or not, the task is treated as a regression task, where the output is a numeric value

between zero and one. This change will be described in more detail in this chapter. For these experiments, the same machine learning system — RDS — is used as for the experiments described in Chapter 5. The same data are also used to train the models and to tune the parameters. The all but last section of the chapter exemplifies the keyword extraction algorithm on a document from the test set.

7.2 REFINEMENTS

In this section, three refinements made to the keyword extraction algorithm will be discussed. The first one concerns how the NP-chunks are extracted from the documents: by removing the initial determiner for the NP-chunks, better performance is achieved. The second alteration is to use a general corpus for calculating the IDF value. Also the weights for the positive examples are set in a more systematic way, to maximise the performance of the combined model.

7.2.1 REFINING THE NP-CHUNK APPROACH

It was noted in Chapter 6 that when extracting NP-chunks, the accompanying determiners are also extracted (per definition), but that determiners are rarely found at the initial position of keywords. This means that the automatic evaluation treats these keywords as misclassified, although they might have been correct without the determiner. For this reason the determiners *a*, *an*, and *the* are removed when occurring in the beginning of an extracted NP-chunks. The results for the runs when extracting NP-chunks with and without these determiners are found in Table 7.1. As can be seen in this table, the recall increases while the precision decreases. However, the high increase in recall leads to an increase in the F-measure from 33.0 to 36.8. In the experiments on the individual classifiers described in Chapter 5, the highest F-measure is achieved by the n-gram approach (33.9). By making this alteration to how the NP-chunks are extracted, the NP-chunk approach obtains the highest F-measure of the individual classifiers.

Table 7.1: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), when extracting NP-chunks with and without the initial determiners* a, an, *and* the. *The highest values are shown in bold.*

Individual	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
With det.	4 788	9.6	1 421	2.8	**29.7**	37.2	33.0
Without det.	7 510	15.0	2 083	4.2	27.7	**54.6**	**36.8**

Table 7.2: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), for the expert combination when the NP-chunks are extracted with and without the initial determiners* a, an, *and* the. *The highest values are shown in bold.*

Majority vote	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
With det.	4 369	8.7	1 558	3.1	35.7	40.8	38.1
Without det.	4 543	9.1	1 643	3.3	**36.2**	**43.1**	**39.3**

The results for the expert combination taking the majority vote and removing the subsumed keywords, using the improved NP-chunk approach can be found in Table 7.2. In the table, these results are compared to the previous majority vote results (presented in Chapter 6). As can be seen in the table, both the precision and the recall increase compared to the previous run, leading to an increase in the F-measure.

7.2.2 USING A GENERAL CORPUS

In the experiments described so far, two of the term selection approaches have models that are bagged (the PoS pattern approach consists of a 20-bagged, and the n-gram approach of a 10-bagged model). In the experiments described further, no experiments are made with bagging; to have one parameter less to tune and evaluate. In Table 7.3, the results for the three non-bagged individual classifiers on the test set are shown. As the NP-chunk approach has its best performance on the validation data when not bagged, the result for this classifier is the same as that found in Table 7.1 (without initial determiner). In Table 7.3, the performance when taking the majority vote of the three single classifiers removing the subsumed terms, is also shown. While the n-gram approach has worse performance, the PoS pattern approach actually performs better on the test data as a single classifier than when bagged (29.1 compared to 28.1). This is not the case on the validation data, and the reason why it is evaluated as a bagged classifier in the previous experiments. This is probably the reason why the combined classifier taking the majority vote of the three single classifiers performs better (40.1), than when combing the bagged classifiers (39.3, see Table 7.2).

In the experiments described in Chapter 5, only the documents present in the training, validation, and test set respectively are used to calculate the inverse document frequency. This means that the collection is rather homogenous. An experiment has been performed where the IDF is instead calculated on a set of 200 arbitrarily chosen documents from the British National Corpus[1] (BNC). The results are presented in Table 7.3, in which 'G.C.' denotes 'General Corpus'. If comparing the results calculating the IDF from the BNC documents in place of the set of abstracts, the F-measure increases for all three individual classifiers, and also for the combined model. For the n-gram approach and the combined model, the precision decreases, while the recall increases for all. In other words, using a more general corpus for this calculation leads to better performance of the keyword extraction from the abstracts.

[1] `http://www.natcorp.ox.ac.uk/`

Table 7.3: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), when calculating the IDF from the abstracts, and from a more general corpus (G.C.). The highest values are shown in bold.*

Classifier	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
n-gram	10 498	21.0	2 168	4.3	20.7	56.8	30.3
n-gram G.C.	11 899	23.8	2 423	4.8	20.4	63.5	30.8
NP-chunk	7 510	15.0	2 083	4.2	27.7	54.6	36.8
NP-chunk G.C	7 991	16.0	2 336	4.7	29.2	61.2	39.6
PoS pattern	10 664	21.3	2 110	4.2	19.8	55.3	29.1
PoS pattern G.C.	12 179	24.4	2 474	4.9	20.3	**64.8**	30.9
Maj. vote	5 550	11.1	1 879	3.8	**33.9**	49.2	40.1
Maj. vote G.C.	6 430	12.9	2 122	4.2	33.0	55.6	**41.4**

7.2.3 SETTING THE WEIGHTS

Since the data set is unbalanced — there are a larger number of negative than positive examples — the positive examples are given a higher weight when training the prediction models. In the experiments described so far, the weights given to the positive examples are those resulting in the best performance for each individual classifier (as described in Section 5.2). For the results presented further, the weights are instead set according to which individual weight that maximises the F-measure for the combined model on the validation set. The weight given to the positive examples for each term selection approach has in a (rather large) number of runs been altered systematically for each classifier, and the combination that results in the best performance is selected. The results on the test set are presented in Table 7.4. As can be seen in this table, the recall decreases, while the precision and the F-measure increase.

Table 7.4: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), when combining the classifiers with the best individual weight and with the best combination respectively. The highest values are shown in bold.*

Majority vote	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
Individual best	6 430	12.9	2 122	4.2	33.0	**55.6**	41.4
Best combined	4 120	8.2	1 648	3.3	**40.0**	43.2	**41.5**

7.3 REGRESSION VS. CLASSIFICATION

In the experiments described so far, the automatic keyword indexing task is treated as a binary classification task, where each candidate term is classified either as a keyword or a non-keyword.

RDS allows for the prediction to be treated as a regression task (Breiman *et al.*, 1984). This means that the prediction is given as a numerical value, instead of a category. When training the regression models, the candidate terms being manually assigned keywords are given the value one, and all other candidate terms are assigned the value zero. In this fashion, the prediction is a value between zero and one, and the higher the value, the more likely a candidate term is to be a keyword (according to the model).

To combine the results from the three models, there are two alternatives. Either the prediction value can be added for all candidate terms, or it can

Table 7.5: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), when using classification and regression. The highest values are shown in bold.*

Learning Task	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
Classification	4 120	8.2	1 648	3.3	**40.0**	43.2	41.5
Regression	5 380	10.8	2 093	4.2	38.9	**54.8**	**45.5**
Reg. ind. thresh.	5 628	11.3	2 089	4.2	37.1	54.7	44.2

be added only if it is over a certain threshold set for each model, depending on the model's individual performance. Regardless, a candidate term may be selected as a keyword even if it is extracted by only one method, provided that the value is high enough. The threshold values are defined based on the performance of the models on the validation data. In Table 7.5, results for two regression runs on the test data are presented. The total number of manually assigned keywords present in abstract in the test data is 3 816, and is the number on which the recall is calculated. On average, 7.6 manually assigned keywords are present per document. These two regression runs are in Table 7.5 compared to the best performing classification run. The first regression run ('Regression') is when all candidate terms having an added value over a certain threshold are selected. The second presented regression run (Regression with individual threshold: 'Reg. ind. thresh.') is when a threshold is set for each individual model: if a prediction value is below this threshold it does not contribute to the added value for a candidate term. In this case, the threshold for the total score is slightly lower than when no individual threshold is set. Both regression runs have a higher F-measure than the classification run, due to the fact that recall increases, more than what the precision decreases. The run without individual thresholds results in the highest F-measure.

It might be interesting to compare the results for the regression with the results when assigning all candidate terms that are selected by the three term selection approaches. This will show how much that is gained by applying prediction models using regression. The results are found in Ta-

Table 7.6: *Results when assigning all candidate terms selected by the three term selection approaches, without any prediction models. In the table is shown: the number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and the F-measure (F). The highest values are shown in bold.*

No prediction	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
Sub. not removed	54 322	108.6	3 675	7.4	6.7	**96.3**	12.6
Sub. removed	24 351	48.7	2 514	5.0	**10.3**	65.9	**17.9**

ble 7.6, and should be compared to 'Regression' in Table 7.5. The number of manually assigned keywords present in the documents is 3 816, and as mentioned, this is the figure with which the recall is calculated. In Table 4.1 the results when assigning all candidate terms for each term selection approach respectively to the training data are presented. The highest recall is then obtained by the PoS pattern approach. Thus, this approach is best at detecting the manual keywords in the documents. Also for the test set, this approach has the highest recall (87.5). As can be seen in Table 7.6, if taking the union of all extracted candidate terms, a recall of 96.3 is achieved. Thus, there are still 141 manually assigned keywords in the documents that none of the three term selection approaches has found.

7.4 DEFINING THE NUMBER OF KEYWORDS

If closer inspecting the best regression run, this combined model assigns on average 10.8 keywords per document. The actual distribution varies from 3 documents with 0, to 1 document with 32 keywords. As mentioned, the prediction value from a regression model is numeric, and indicates how likely a candidate term is to be a keyword. It is thus possible to rank the output, and consequently to limit the number of keywords assigned per document. A set of experiments has been performed with the aim to find what number

Table 7.7: *The number of assigned (Assign.) keywords in total and mean per document; the number of correct (Corr.) keywords in total and mean per document; precision (P); recall (R); and F-measure (F), when assigning all terms over the threshold (All), and limiting the number of terms assigned per document (Max. 12, and 3–12 respectively). The highest values are shown in bold.*

Regression	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	P	R	F
All	5 380	10.8	2 093	4.2	38.9	**54.8**	**45.5**
Max. 12	4 290	8.6	1 784	3.6	**41.6**	46.8	44.0
3–12	4 309	8.6	1 789	3.6	41.5	46.9	44.0

of keywords per document that results in the highest F-measure, by varying the number of keywords assigned. In these experiments, only terms with an added value over the empirically set threshold are considered, and the candidate terms with the highest values are selected first. The best performance is when a maximum of twelve keywords is selected for each document. (The subsumed terms are removed after that the maximum number of keywords is selected.) As can be seen in Table 7.7 ('All' compared to 'Max. 12'), the F-measure decreases as does the recall, although the precision increases, when limiting the number of keywords.

There are, however, still some documents that do not get any selected keywords. To overcome this, three terms are assigned to each document even if the added regression value is below the threshold. Doing this gives a slightly lower precision, while the recall increases slightly. The F-measure is unaffected (see Table 7.7: 3–12).

7.5 EXEMPLIFYING THE EXTRACTION

An example of the automatic keyword extraction will now be given. For this purpose, an abstract from the test set has been selected (quite randomly), and is presented in Figure 7.1. In Figures 7.2–7.4, all candidate terms

extracted from this abstract by each term selection approach are displayed. The terms are sorted in order of "keywordness", as judged by each trained regression model. The n-gram approach (Figure 7.2) extracts in total 167, the NP-chunk approach (Figure 7.3) 42, and the PoS pattern approach (Figure 7.4) 129 candidate terms from this abstract. The abstract consists of 191 tokens. In Figure 7.5, the automatically assigned keywords for the abstract are displayed, that is, all candidate terms with an added regression value over the threshold, with the subsumed terms removed; in this case nine terms. Finally, in Figure 7.6, the manually assigned keywords that are present in the abstract are presented.

Design and implementation of a 3-D mapping system for highly irregular shaped objects with application to semiconductor manufacturing. The basic technology for a robotic system is developed to automate the packing of polycrystalline silicon nuggets into fragile fused silica crucible in Czochralski (melt pulling) semiconductor wafer production. The highly irregular shapes of the nuggets and the packing constraints make this a difficult and challenging task. It requires the delicate manipulation and packing of highly irregular polycrystalline silicon nuggets into a fragile fused silica crucible. For this application, a dual optical 3-D surface mapping system that uses active laser triangulation has been developed and successfully tested. One part of the system measures the geometry profile of a nugget being packed and the other the profile of the nuggets already in the crucible. A resolution of 1 mm with 15-KHz sampling frequency is achieved. Data from the system are used by the packing algorithm, which determines optimal nugget placement. The key contribution is to describe the design and implementation of an efficient and robust 3-D imaging system to map highly irregular shaped objects using conventional components in context of real commercial manufacturing processes.

Figure 7.1: *An abstract from the test set.*

irregular shaped objects	3-d surface mapping
polycrystalline silicon nuggets	active laser triangulation
shaped objects	highly irregular
fused silica crucible	highly irregular shaped
silicon nuggets	basic technology
surface mapping system	manufacturing processes
fragile fused silica	packing algorithm
commercial manufacturing processes	mapping
irregular polycrystalline silicon	czochralski
optical 3-d surface	resolution of mm
packing constraints	sampling
fused silica	15-khz
polycrystalline silicon	commercial manufacturing
semiconductor manufacturing	3-d mapping system
semiconductor wafer	system for highly
silica crucible	automate
wafer production	frequency
semiconductor wafer production	3-d imaging
geometry profile	crucible
laser triangulation	semiconductor
nugget placement	silica
sampling frequency	system
surface mapping	silicon
imaging system	3-d
mapping system	polycrystalline
conventional components	algorithm
3-d imaging system	geometry
optimal nugget placement	mm
3-d surface	placement
active laser	triangulation
challenging task	packing
delicate manipulation	implementation
robotic system	components
key contribution	constraints
real commercial manufacturing	data
robust 3-d imaging	objects

processes	pulling
laser	successfully tested
manipulation	conventional
resolution	efficient
context	robust
imaging	system measures
15-khz sampling	dual optical
15-khz sampling frequency	irregular polycrystalline
crucible in czochralski	optical 3-d
developed and successfully	real commercial
highly irregular polycrystalline	robust 3-d
map highly	developed
map highly irregular	contribution
melt pulling	challenging
mm with 15-khz	delicate
nugget being packed	3-d mapping
objects using	achieved
shaped objects using	active
using conventional	application
using conventional components	application to semiconductor
melt	automate the packing
robotic	basic
design	commercial
objects with application	components in context
manufacturing	context of real
surface	describe
production	describe the design
profile	design and implementation
task	determines
technology	determines optimal
fragile fused	determines optimal nugget
irregular shaped	developed to automate
dual	difficult
fragile	difficult and challenging
fused	dual optical 3-d
optical	efficient and robust

```
frequency is achieved       packing of highly
highly                      packing of polycrystalline
irregular                   real
key                         requires
manipulation and packing    requires the delicate
measures                    shaped
measures the geometry       successfully
nuggets                     system is developed
nuggets into fragile        system to map
objects using conventional  tested
optimal                     wafer
optimal nugget
```

Figure 7.2: *Candidate terms; the* n*-gram approach.*

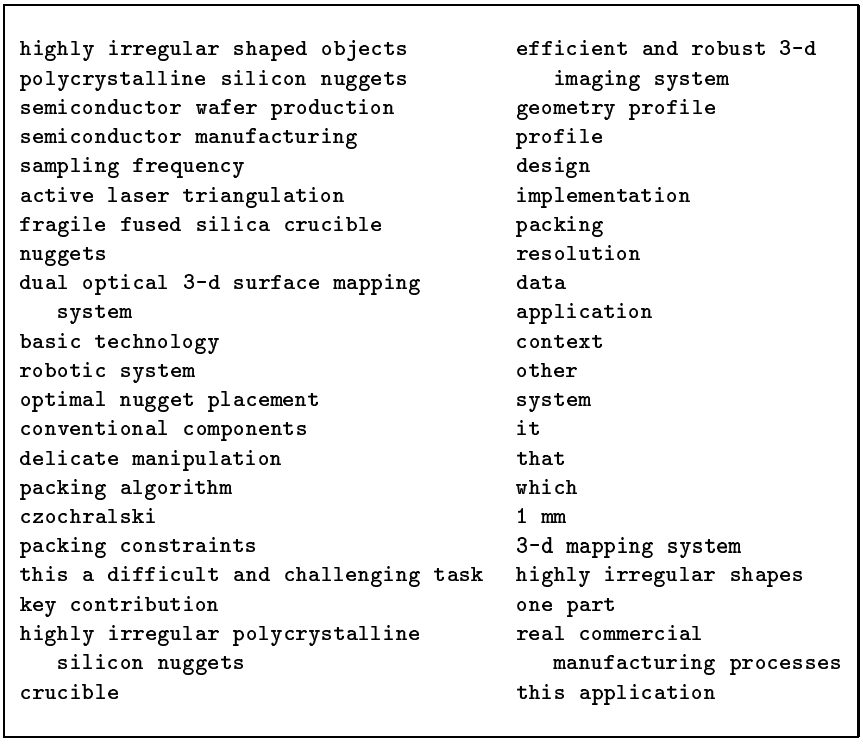

Figure 7.3: *Candidate terms for the NP-chunk approach.*

polycrystalline silicon nuggets	semiconductor wafer
shaped objects	surface mapping
dual optical 3-d surface	wafer production
fragile fused silica	sampling frequency
fragile fused silica crucible	3-d imaging
fused silica crucible	3-d surface
irregular shaped objects	active laser
silicon nuggets	basic technology
3-d surface mapping system	commercial manufacturing
optical 3-d surface	delicate manipulation
highly irregular shapes	key contribution
silica crucible	optimal nugget
3-d surface mapping	robotic system
active laser triangulation	objects
semiconductor manufacturing	packing constraints
semiconductor wafer production	surface mapping system
mapping system	packing algorithm
optical 3-d surface mapping	pulling semiconductor
3-d imaging system	czochralski
commercial manufacturing processes	geometry
conventional components	mm
imaging system	semiconductor
nugget placement	silica
optimal nugget placement	triangulation
real commercial manufacturing	wafer
real commercial manufacturing processes	challenging task
robust 3-d imaging	algorithm
robust 3-d imaging system	crucible
fused silica	frequency
polycrystalline silicon	placement
irregular polycrystalline silicon	silicon
irregular polycrystalline silicon nuggets	components
geometry profile	mapping
laser triangulation	sampling
	laser
	1 mm

3-d mapping	irregular polycrystalline
application	irregular shaped
context	optical 3-d
data	manipulation
implementation	active
manufacturing	basic
packing	challenging
profile	commercial
surface	conventional
design	delicate
system	difficult
3-d	efficient
polycrystalline	key
nugget being	real
processes	robotic
contribution	robust
imaging	shaped
production	15-khz
task	being
technology	constraints
dual	manufacturing processes
fragile	nuggets
fused	one part
irregular	other
optical	part
optimal	pulling semiconductor wafer
resolution	real commercial
pulling	robust 3-d
dual optical	using
fragile fused	

Figure 7.4: *Candidate terms; the PoS pattern approach.*

```
polycrystalline silicon nuggets
irregular shaped objects
semiconductor wafer production
semiconductor manufacturing
fragile fused silica crucible
active laser triangulation
sampling frequency
optimal nugget placement
conventional components
```

Figure 7.5: *Automatically assigned keywords using regression, with subsumed terms removed.*

```
robust 3-D imaging system
packing algorithm
commercial manufacturing processes
irregular shaped objects
sampling frequency
highly irregular polycrystalline silicon nuggets
fragile fused silica crucible
highly irregular shaped objects
semiconductor manufacturing
active laser triangulation
robotic system
polycrystalline silicon nuggets
```

Figure 7.6: *Manually assigned keywords (present in abstract).*

7.6 DISCUSSION

In this chapter, a number of experiments leading to better performance of the keyword extraction algorithm have been presented. One improvement concerns how the NP-chunks are extracted, where the results are improved by excluding the initial determiners *a*, *an*, and *the*. Possibly, this improvement could be yet higher if all initial determiners were removed from the NP. Another improvement concerns how the IDF is calculated, where the F-measure of the classifiers increases when a general corpus is used. A third improvement concerns how the weights to the positive examples are set. By adjusting the weights to maximise the performance of the combined model, the F-measure increases. Also, one major change is made to the algorithm, as the learning task is redefined. This is done by using regression instead of classification for the machine learning. Apart from an increase in performance by regression, this enables a ranked output of the keywords. This in turn makes it easy to vary the number of keywords selected per document, in case necessary for some types of applications. In addition, compared to classification, regression resembles reality in the sense that some words are definitely keywords, some are definitely not, but there are also many candidate terms that are keywords to a certain extent. Thus, there is a continuum of the candidate terms' "keywordness". Compared to the experiments on automatic keyword extraction presented by Turney (2000) and Frank *et al.* (1999), the number of keywords extracted per document varies depending on the threshold value, and is not identical for all documents in the set. This is also similar to how manual indexing is performed.

If looking at the inter-judgement agreement between the keywords selected by the combined model assigning no more than twelve keywords per document and the manually assigned keywords for the documents in the test set, it is 28.2%. (This is the intersection of selected keywords between the system and the human, divided by the union of the two.) Thus the performance of the keyword extraction algorithm is at least as consistent as that of inexperienced professional indexers (which is 20%, according to Bureau van Dijk (1995)). This is, however, only true to a certain extent, as some of the keywords selected by the automatic extractor would never have been considered by a human — not even a non-professional[2].

[2]Two examples of such keywords from the test data would be 'As luck' and 'Comprehension goes'.

Chapter 8

Integration with a search engine

This chapter exemplifies how automatically extracted keywords can yield added value to an information retrieval task. This is achieved by integrating the keyword extraction algorithm with a search engine, where the keywords present more information about the Web pages returned as a response to the queries. The keyword extraction consists of three regression models that are combined to decide what words or sequences of words in the texts that are suitable as keywords. The models, in turn, are built using different definitions of what constitutes a term in a written document.

8.1 Introduction

When searching a collection for relevant documents by means of a search engine, a user states a number of query words. These words are then matched with the documents in the collection. The common way to present the result is by showing a list of the hits in order of estimated relevance. If searching the Web using Google, the page displaying the results presents for each document, among other things, the title; snippets of the query

words in context; a manual description from the Google directory (when such exists); and the Web page's URL. This chapter describes a Web application — called Keegle — where the search results from Google are extended with keywords automatically extracted from the content of each displayed hit[1]. In addition, an example of a result page that also displays keywords extracted automatically from each document shown in the hit list, is given.

8.2 SYSTEM DESCRIPTION

Google provides an API for searching the Google Web index, and retrieving cached Web pages[2]. The Web front-end of Keegle is implemented in Java, from which the calls to Google and to the external programs are made. The keyword extractor is implemented in Perl, the partial parser in C++, while the prediction models are in Prolog, and executed by RDS.

In Figure 8.1, an overview is given of Keegle. In more detail, the interaction with the system is as follows: a user writes a number of query words into a Web page. The query is received by Keegle, and the request is sent to Google. The documents corresponding to the search hits are retrieved from Google's cache. Thereafter the HTML-tags are removed from the documents, as are lines containing no more than one token (these lines are often Web page menus). In addition, lines containing "Web specific" information such as 'links' are removed. Next, the documents' length is restricted (this action is described further in Section 8.3: Optimisation). The documents are thereafter tagged and chunked using LT CHUNK. The next step is to extract the candidate terms according to the three term selection approaches: n-*grams* corresponds to extracting all uni-, bi- and trigrams leaving out those that begin or end with a stop word. The second method (*NP-chunks*) extracts all NP-chunks, without the initial determiners, as described in Section 7.2.1. The third method (*PoS patterns*) extracts all words or sequences of words that match any of 56 empirically defined PoS tag patterns (PoS tag patterns that are common for manually assigned keywords). (See Chapter 4 for more details on the term selection approaches.) Thereafter the features are calculated for each candidate term, that is, the term

[1] This Web application is not publicly available, due to licence restrictions of the API:s.
[2] `http://www.google.com/apis/`

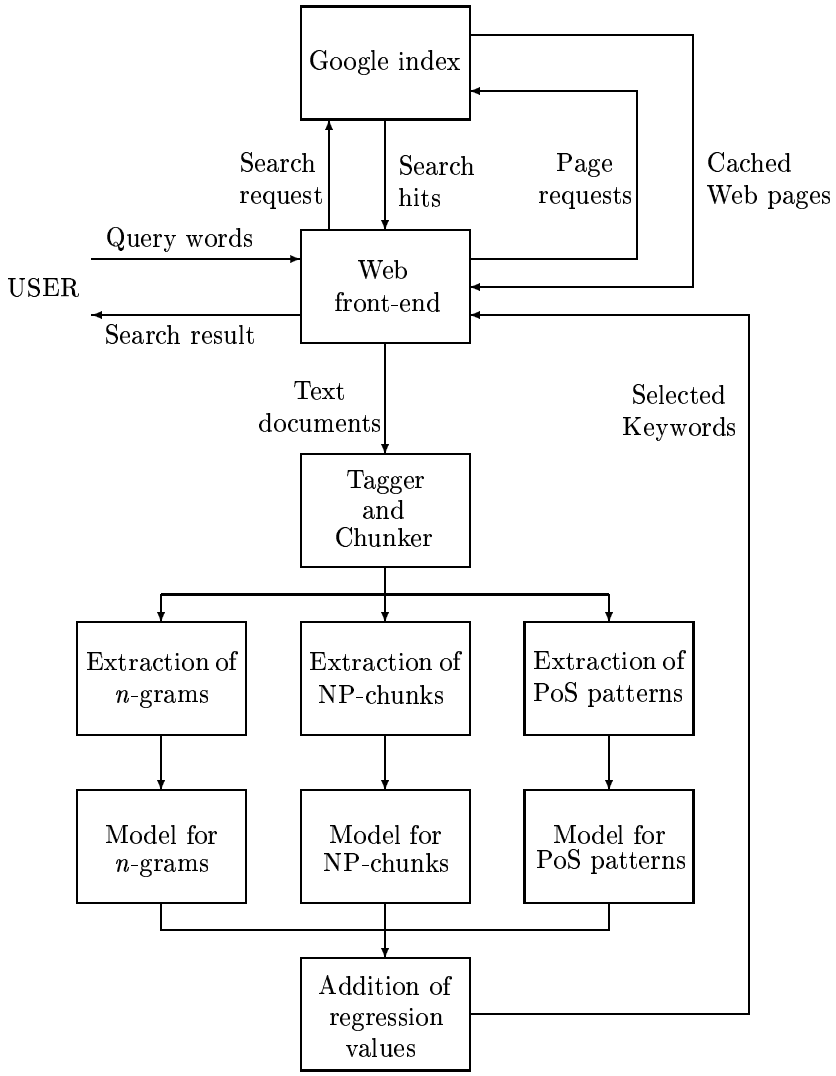

Figure 8.1: *An overview of Keegle: a system for automatic keyword extraction from Web pages.*

frequency, the inverse document frequency, the relative position of the first occurrence, and the PoS tag or tags assigned to the term. Subsequently, the three prediction models assign a value to each candidate term, based on the feature values of the term. Next, the terms are ranked based on the added value given by the three models. At least three keywords are selected for each Web page, but as long as the added value is over a certain threshold, a maximum of twelve keywords may be presented (as described in Section 7.4). As an all but last step, subsumed keywords are removed, and the most frequently occurring unstemmed form in a candidate term's corresponding document is looked up. Finally, the keywords are presented in order of estimated "keywordness", together with most of the information normally presented for a Google search.

An example of what the result can look like is shown in Figure 8.2. If closer inspecting the first hit shown in the result page, this Web page consists of 334 tokens. Extracting all n-grams gives 163 candidate terms in total: 87 unigrams, 43 bigrams, and 33 trigrams. There are 142 words or sequences of words matching the PoS tag patterns, and there are 88 NP-chunks extracted.

8.3 OPTIMISATION

The only characteristic that Web pages share with the scientific journal paper abstracts on which the models have been trained is that Web pages are usually rather short. The length of the abstracts in the training data varies from 15 to 512 tokens. The length of the Web pages for which keywords are to be extracted by Keegle is limited to 1 000 tokens, that is, only the first 1 000 tokens are considered. This is, however, a decision that has more to do with processing time than with the quality of the extracted keywords. The average time it takes from sending a request to seeing the result is with restricted length still more than one minute when extracting terms for the first five Web pages; and that would, of course, be too long when doing a Web search. If automatically extracted keywords were to be a useful feature in a search engine, the keyword indexing would not be done at search time, but in batch when building the index database. The implementation of Keegle has a caching function that saves the URL and the extracted

Figure 8.2: *A screen dump of a result page presenting keywords for each search hit.*

keywords in a look-up table. This means that subsequent searches with the same query are considerable faster (around a second). Apart from restricting the length and the caching function, no other measures have been taken to improve the processing time. One example of an improvement that could be made is to run the external programs as servers to avoid long loading times.

8.4 DISCUSSION

This chapter has discussed Keegle — a Web application where the search hits from Google are extended to include automatically extracted keywords from each Web page presented in the result page. As the models making the prediction of what terms in the documents that are suitable as keywords use regression, the selected keywords can be ranked and presented in order of estimated "keywordness".

Using Google means that the documents from which keywords are to be extracted can be any documents; that they are not limited to the specific type on which the keyword extractor has been trained. Given the difficult task of extracting keywords from unrestricted texts — it is already difficult if the texts are domain specific — the keyword extraction seems to be surprisingly accurate. The performance of Keegle needs, however, to be systematically evaluated. Keegle is developed for English, and keywords selected from Web pages in other languages are, naturally, of a poorer quality. It should also be noted that when doing this kind of keyword indexing, it would not be meaningful to limit the keywords to a set of "allowed" terms, as the selected keywords would be too confined.

Presenting keywords in the result page from a search engine may help the user to quicker judge the relevance of the returned documents. A study performed by Rath *et al.* (1961) shows that presenting documents with their titles only, makes the users judge irrelevant documents as relevant. This risk can be limited by also presenting automatically extracted keywords. Experiments performed by Jing *et al.* (1998) indicate that in comparison to longer summaries, keywords may be just as informative but that judging the relevance requires less time for a human reader. (The comparison

is, however, not pursued, as the focus is on evaluating the performance of summarisation systems.) Another application where the keywords had this purpose was proposed by Hulth *et al.* (2001b). In that application automatically identified keywords were to serve as condensed versions of short news items. The extracted keywords were to be displayed on hand held computers, so that the user of the device easier could decide what items that would be worthwhile to download. Another Web service that could benefit from automatically extracted keywords is CiteSeer[3], where the keywords could give additional information about the presented abstracts.

The research in automatic text retrieval aims at facilitating the accessibility of digital information to its users. Using the keywords as a dense summary for a single document — which is the function they serve in Keegle — is only one way how keywords can be used in an information retrieval situation. They can also be the entrance to a document collection, similar to a back-of-book index. They can be used to cluster retrieved documents before the documents are presented to a user. Or, the keywords can be used to re-rank the hits, where documents having the query words as highly ranked keywords are presented first. Also, the keywords for the top x ranked documents can instead of acting as a summary be listed at the top of the result page, to support the user in refining the query. Integrating the keywords in various manners also enables different evaluation settings, as evaluations with users will play an important role in further developing the keyword extraction algorithm, and to tune it to different information retrieval tasks.

[3] `http://citeseer.ist.psu.edu/`

Chapter 9

Summary and conclusions

This last chapter of the dissertation consists of three sections. The first section summarises the algorithm for automatic keyword extraction. The second section describes concrete ideas for future work, and the third and last section presents concluding remarks.

9.1 An algorithm for keyword extraction

Section 2.1 lists six questions concerning manual keyword indexing (from O'Connor (1996)). These questions are just as applicable as it comes to automating the procedure. They will now be revisited and answered in the light of the performed research. In doing so, the algorithm for automatic keyword extraction that is the result of the research presented in the dissertation will be outlined. (The order of the questions is slightly modified, for the sake of clarity.)

9.1.1 RESOURCES

The keyword extraction algorithm requires a number of resources for training the prediction models, and subsequently for automatically selecting keywords to documents that are to be indexed. The algorithm is developed for English. Consequently the required text collections should be written in English, and the external tools should also be developed for the English language. The required resources are as follows:

- A collection of documents with manually assigned keywords
- A general corpus
- A part-of-speech tagger
- A noun phrase chunker

9.1.2 PROCEDURE

In answering the question from O'Connor (1996) listed first in this section, the main characteristics of the algorithm are described.

- Which terms should be extracted?

  Pre-process (or represent) the data:

  - Extract the candidate terms, that is, extract:
    - all uni-, bi, and trigrams that do not begin or end with a stop word
    - all NP-chunks as judged by the chunker, excluding the initial determiners *a*, *an*, and *the*
    - all PoS tagged words or sequences of words matching any of the set of empirically defined PoS patterns (frequently occurring patterns of manual keywords)
  - Case-fold and stem the candidate terms

- Calculate the features:
  * term frequency
  * inverse document frequency, using a general corpus
  * position of the first occurrence, normalised by length
  * the most frequent PoS tag or tags assigned to the term

Train the models:

- Assign the candidate terms that are equivalent to a stemmed manual keyword the value one, and all others the value zero.
- Train the models using a learning algorithm that outputs a regression value.

Apply the prediction models:

- Pre-process the documents for which keywords are to be selected as described above. Let the three trained models predict the values of the candidate terms for the three term selection approaches respectively.
- Add the regression values from the three models, for each unique candidate term.
- Select the candidate terms that meet the criteria given in the answer for the next question ("How many terms should be extracted").
- Remove terms that are subsumed by another keyword selected for the document.
- For each selected keyword, present the most frequently occurring unstemmed form in the document.

• How many terms should be extracted?

In case there is no need to limit the number of keywords assigned per document, select all candidate terms having an added regression value over the empirically defined threshold. If, on the other hand, the number of terms should be limited, extract at a maximum twelve keywords per document, provided that the added regression value is higher than the empirically defined threshold value.

- In what order should they be presented?

Present the keywords in order of estimated "keywordness", based on the added regression value given by the three prediction models.

- Should the terms be derived rather than extracted?

The approach taken in this work is extraction. To automatically assign keywords that are not present in a document is a different, albeit challenging research question, and is saved for future work. The derivation can be done on different levels: it may mean that the base form and not an inflected form found in a document is presented to the user (for example plural is altered to singular). Different word classes of a common root can also be merged into one class. It can also mean that a synonym is used for a term in a document, or that a more general term is given for a set of more specialised concepts (for example *fruits* instead of *plums*, *pears* and *papayas*).

- Should concepts be generalised?

This has not been touched upon in the work presented in the dissertation. It is however an interesting issue for further research, and is also related to the previous question.

- What rules should guide the extraction?

As stated in Section 3.3, one advantage with rule induction is that a trained prediction model is in a form that may be examined, and also understandable to a human. This is however only true to a certain extent. The three regression models that are used for Keegle (see Chapter 8) consist of 713 (n-grams), 870 (NP-chunks), and 416 (PoS patterns) rules respectively. To get an overview of such a large number of rules is, of course, difficult. Also, if a human were to write a number of rules for keyword extraction, the rules would clearly differ from those produced by a learning algorithm. (See Section 3.3 for an example of automatically produced rules.) Nonetheless, the rules might give an insight into how the predictions are made.

9.2 FUTURE WORK

This all but last section of the dissertation will summarise the most important issues that have not been dealt with so far.

The experiments described in the dissertation are performed on abstracts, that is, the models are trained and evaluated on abstracts. The gain of that is two-fold: many databases consist of abstracts only; and there is less noise in the input to the learning algorithm. However, the feature values would possibly convey more information if the texts were longer. The next important issue is to compare how the extraction algorithm would perform on full texts. Also, the described experiments are all performed on the same data set. It would be valuable to train and evaluate the algorithm on another data set, still consisting of abstracts. One alternative could be to compare the keyword extraction from full texts to that from abstracts using the same documents.

The algorithm is developed on English texts for the English language. An interesting aspect of further research is to examine how the discussed method may be applied to other languages, and how much of the algorithm that may be generalised to be valid for other languages as well.

If looking at the application as it is implemented at the time of writing, it has one small albeit striking short-coming, namely the lack of correct capitalisation. The current implementation only capitalises the first letter of the first token of each keyword. This can partly be solved by looking at the capitalisation in the document, but in order to not wrongly capitalise keywords that are not proper nouns and that happen to occur in the beginning of a sentence, a more sophisticated method is needed.

Another issue to focus on in the nearer future is the evaluation. Apart from performing the evaluations using human judges at the time of evaluation, as described in Section 3.4, it would also be valuable to further investigate the indexing consistency among people (or rather the lack thereof). Especially it could be interesting to study how the indexing changes for one person over time.

In order to enable researchers in the area to make fair comparisons of developed algorithms, a common corpus for testing is required. Such a corpus

should contain documents where each document has keywords assigned by several professional indexers. Also, the corpus should be of different genres.

Although this dissertation focuses on written documents, it is easy to foresee that keywords need to be applied to all types of media. This is one important way to pursue this research, and doing so will present other challenges due to the multi-medial character of the information. Future work should also go in the direction of generating (as opposed to extracting) keywords, by for example exploring the potential knowledge provided by a thesaurus.

One advantage of automatic indexing is that keywords do not have to be assigned once and for all — as is normally the case with manual indexing. On the contrary, it can be a dynamic process where consideration is taken to the user, the task, and the time. In other words, different keywords may be assigned to the same document at different occasions (or at the same time for different purposes). In order to select keywords that are tailored for specific users, their vocabulary as well as preferences need to be included in the process. Thus, the task of selecting keywords should be more interactive. In the work done so far, terms that correspond to a document's subject, with no regard to the varying meaning (as discussed in Section 2.1) are assigned. The next challenge is to assign keywords where consideration is taken to the users and their specific information needs, that is, keywords reflecting the meaning of the documents. In assigning the keywords dynamically, the evaluation becomes even more challenging, and yet higher demands will be posed on the evaluation process.

9.3 CONCLUDING REMARKS

The research described in this dissertation concerns automatic keyword indexing, where the goal is to automatically find a small set of terms describing the content of a document. More specifically, the dissertation concerns keyword extraction, in which the keywords are present verbatim in the document for which they are selected. The approach taken is that of supervised machine learning, that is, prediction models are constructed by training a learning algorithm on documents with known keywords. These models are

in turn applied to the documents for which keywords are to be extracted. The performance of the prediction models is evaluated by calculating the F-measure for the selected keywords, with equal weight given to the precision and the recall.

To not let the selection of candidate terms be an arbitrary process — which is the case when extracting n-grams — and better capture the idea of keywords having certain linguistic properties, two alternative approaches to select the candidate terms are explored. These are to extract all NP-chunks, and to extract all terms matching any of a number of empirically defined PoS patterns. Thus, these two term selection approaches are more linguistically oriented. The best individual performance is achieved by the NP-chunk approach. In the work, four features are explored: term frequency; inverse document frequency; relative position of the first occurrence; and the PoS tag or tags assigned to the candidate term. The PoS tag feature turns out to dramatically improve the performance of the models, independently of term selection approach applied. The best performance is, however, achieved when the predictions of all three models are combined.

This dissertation has shown that by incorporating linguistic knowledge into the representation constituting the basis for a learning algorithm, a more accurate keyword extraction is achieved. Automatic keyword extraction is thus an application in which linguistically oriented methods outperform those based purely on standard statistical methods. For this reason it would be an ideal test bed for exploring the potential of natural language processing tools for solving problems concerning natural language tasks.

REFERENCES

Asker, L. and Maclin, R. 1997. Ensembles as a sequence of classifiers. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI'97)*, Nagoya, Japan.

Baeza-Yates, R. and Ribeiro-Neto, B. 1999. *Modern Information Retrieval.* Addison Wesley.

Bahler, D. and Navarro, L. 2000. Methods for combining heterogeneous sets of classifiers. In *17th National Conference on Artificial Intelligence (AAAI 2000): Workshop on New Research Problems for Machine Learning.*

Banko, M., Mittal, V. O., and Witbrock, M. J. 2000. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics.*

Barker, K. and Cornacchia, N. 2000. Using noun phrase heads to extract document keyphrases. In *Canadian Conference on AI.*

Basili, R., Moschitti, A., and Pazienza, M. T. 2002. Empirical investigation of fast text categorization over linguistic features. In *Proceedings of the 15th European Conference on Artificial Intelligence (ECAI 2002)*, pp. 485–489.

Boguraev, B. and Kennedy, C. 1999. Applications of term identification technology: Domain description and content characterisation. *Natural Language Engineering*, **5**(1):17–44.

Bourigault, D., Jacquemin, C., and L'Homme, M.-C., editors. 2001. *Recent Advances in Computational Terminology.* John Benjamins Publishing Company, Amsterdam.

Brandow, R., Mitze, K., and Rau, L. F. 1995. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management*, **31**(5):675–685.

Breiman, L. 1996. Bagging predictors. *Machine Learning*, **24**(2):123–140.

Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. 1984. *Classification and Regression Trees.* Chapman & Hall, New York.

Brown, J. S. and Duguid, P. 2000. *The Social Life of Information.* Harvard Business School Press.

Bureau van Dijk. 1995. Parlement Européen, Evaluation des opérations pilotes d'indexation automatique (Convention spécifique no 52556), Rapport d'évalution finale.

Carroll, J. M. and Roeloffs, R. 1969. Computer selection of keywords using word-frequency analysis. *American Documentation.*

Daille, B., Gaussier, E., and Langé, J.-M. 1994. Towards automatic extraction of monolingual and bilingual terminology. In *Proceedings of COLING-94*, pp. 515–521, Kyoto, Japan.

Dexter, M. E. and Zunde, P. 1969. Indexing consistency and quality. *American Documentation*, **20**(4):259–267.

Dietterich, T. G. 1998. Machine learning research: Four current directions. *The AI Magazine*, **18**(4):97–136.

Earl, L. L. 1970. Experiments in automatic extracting and indexing. *Information Storage & Retrieval*, **6**:313–334.

Edmundson, H. P. 1969. New methods in automatic abstracting. *Journal of the Association for Computing Machinery*, **16**(2):264–285.

Evans, D. K., Klavans, J. L., and Wacholder, N. 2000. Document processing with LinkIT. In *Proceedings of the RIAO Conference*, Paris, France.

Ferber, R. 1997. Automated indexing with thesaurus descriptors: A co-occurrence based approach to multilingual retrieval. In *Proceedings of Research and Advanced Technology for Digital Libraries. First European Conference (ECDL'97)*, volume 1324 of *Lecture Notes in Computer Science*, pp. 233–252. Springer.

Fox, C. 1992. Lexical analysis and stoplists. In Frakes, W. B. and Baeza-Yates, R., editors, *Information Retrieval: Data Structures & Algorithms*, pp. 102–130. Prentice-Hall, New Jersey.

Frank, E., Paynter, G. W., Witten, I. H., Gutwin, C., and Nevill-Manning, C. G. 1999. Domain-specific keyphrase extraction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'99)*, pp. 668–673, Stockholm, Sweden.

Gaizauskas, R. and Wilks, Y. 1998. Information extraction: Beyond document retrieval. *Journal of Documentation*, **54**(1):70–105.

Harabagiu, S. and Moldovan, D. 2003. Question answering. In Mitkov (2003), pp. 560–582.

Hodge, G. M. 1992. *Automated Support to Indexing.* The National Federation of Abstracting and Information Services, Philadelphia, PA.

Hulth, A. 2001. *The Gist of Written Documents: Automatic Derivation and Evaluation of Content Descriptors.* Licentiate Thesis, Department of Computer and Systems Sciences, Stockholm University.

Hulth, A. 2003a. Analysing term selection approaches and features for automatic keyword extraction. In *Proceedings of the 14th Nordic Conference on Computational Linguistics (NoDaLiDa).* Forthcoming.

Hulth, A. 2003b. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2003)*, pp. 216–223, Sapporo, Japan. Association for Computational Linguistics.

Hulth, A. 2004a. Reducing false positives by expert combination in automatic keyword indexing. In Nicolov, N., Botcheva, K., Angelova, G., and Mitkov, R., editors, *Recent Advances in Natural Language Processing III: Selected Papers from RANLP 2003*, Current Issues in Linguistic Theory (CILT). John Benjamins, Amsterdam/Philadelphia. Forthcoming.

Hulth, A. 2004b. Enhancing linguistically oriented automatic keyword extraction. In *Proceedings of the Human Language Technology Conference/North American Chapter of the Association for Computational Linguistics Annual Meeting*, Boston, USA. Forthcoming.

Hulth, A., Karlgren, J., Jonsson, A., Boström, H., and Asker, L. 2001a. Automatic keyword extraction using domain knowledge. In Gelbukh, A., editor, *Proceedings of Second International Conference on Computational*

Linguistics and Intelligent Text Processing, volume 2004 of *Lecture Notes in Computer Science*, pp. 472–482, Mexico City, Mexico. Springer.

Hulth, A., Olsson, F., and Tierney, M. 2001b. Exploring key phrases for browsing an online news feed in a mobile context. In *Proceedings of ECSQARU-2001 Workshop: Management of Uncertainty and Imprecision in Multimedia Information Systems*, Toulouse, France.

Jacquemin, C., Daille, B., Royauté, J., and Polanco, X. 2002. In vitro evaluation of a program for machine-aided indexing. *Information Processing and Management*, **38**(6):765–792.

Jing, H., Barzilay, R., McKeown, K., and Elhadad, M. 1998. Summarization evaluation methods: Experiments and analysis. In *AAAI Intelligent Text Summarization Workshop*, pp. 60–68, Stanford, CA.

Justeson, J. S. and Katz, S. M. 1995. Technical terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, **1**(1):9–27.

Kageura, K., Yoshioka, M., Koyama, T., and Nozue, T. 1998. Towards a common testbed for corpus-based computational terminology. In *Proceedings of the 1st Workshop on Computational Terminology (Computerm'98)*, pp. 81–85.

Klavans, J. L., McKeown, K. R., Kan, M.-Y., and Lee, S. 1998. Resources for evaluation of summarization techniques. In *Proceedings of the 1st International Conference on Language Resources and Evaluation*, Grenada, Spain.

Kraaij, W., Spitters, M., and Hulth, A. 2002. Headline extraction based on a combination of uni- and multidocument summarization techniques. In *Proceedings of the Workshop on Automatic Summarization and Document Understanding Conference (DUC 2002)*, volume 2, pp. 15–29, Philadelphia, Pennsylvania, USA.

Lahtinen, T. 2000. *Automatic Indexing: An Approach Using an Index Term Corpus and Combining Linguistic and Statistical Methods.* Ph.D. thesis, Department of general linguistics, University of Helsinki.

Lancaster, F. W. 1998. *Indexing and Abstracting in Theory and Practice.* Library Association Publishing, London, 2nd edition.

Langridge, D. W. 1989. *Subject Analysis: Principles and Procedures.* Bowker-Saur.

Luhn, H. P. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, **1**:309–317.

Luhn, H. P. 1959. Auto-encoding of documents for information retrieval systems. In Boaz, M., editor, *Modern Trends in Documentation*, pp. 45–58. Pergamon Press, London.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. 1994. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics*, **19**(2):313–330.

Mitchell, T. M. 1997. *Machine Learning.* McGraw-Hill.

Mitkov, R., editor. 2003. *The Oxford Handbook of Computational Linguistics.* Oxford University Press.

Montejo Ráez, A. 2002. Towards conceptual indexing using automatic assignment of descriptors. In *Workshop in Personalization Techniques in Electronic Publishing on the Web: Trends and Perspectives*, Málaga, Spain.

Nienhuys-Cheng, S.-H. and de Wolf, R. 1997. *Foundations of Inductive Logic Programming.* Springer.

O'Connor, B. C. 1996. *Explorations in Indexing and Abstracting: Pointing, Virtue, and Power.* Library and Information Science Text Series. Libraries Unlimited, Inc.

Porter, M. 1980. An algorithm for suffix stripping. *Program*, **14**(3):130–137.

Pouliquen, B., Steinberger, R., and Ignat, C. 2003. Automatic annotation of multilingual text collections with a conceptual thesaurus. In *Proceedings of the Workshop on Ontologies and Information Extraction held as part of the Eurolan 2003 summer school on The Semantic Web and Language Technology — Its Potential and Practicalities*, pp. 19–28, Bucharest, Romania.

Radev, D. R., Hovy, E., and McKeown, K. 2002. Introduction to the special issue on summarization. *Computational Linguistics*, **28**(4):399–408.

Radev, D. R., Teufel, S., Saggion, H., Lam, W., Blitzer, J., Qi, H., Çelebi, A., Liu, D., and Drabek, E. 2003. Evaluation challenges in large-scale document summarization. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pp. 375–382, Sapporo, Japan.

Rath, G. J., Resnik, A., and Savage, T. R. 1961. Comparisons of four types of lexical indicators of content. *American Documentation*, pp. 126–130.

RDS. 2003. Rule Discovery System, Compumine AB. `www.compumine.com`.

Salton, G. and McGill, M. J. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Publishing Company.

Siegel, S. and Castellan, N. J. J. 1988. *Non-Parametric Statistics for the Behavioral Sciences*. McGraw-Hill, New York.

Sparck Jones, K. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, **28**:11–21.

Sparck Jones, K. 1999. What is the role of NLP in text retrieval? In Strzalkowski, T., editor, *Natural Language Information Retrieval*, chapter 1, pp. 1–21. Kluwer Academic Publishers.

Sparck Jones, K. and Galliers, J. R. 1996. *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer.

Strzalkowski, T., Perez-Carballo, J., Karlgren, J., Hulth, A., Tapanainen, P., and Lahtinen, T. 1999. Natural language information retrieval: TREC-8 report. In Voorhees, E. M. and Harman, D. K., editors, *Proceedings of the Eighth Text REtrieval Conference (TREC 8)*, NIST Special Publication 500-246, pp. 381–390.

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

Tzoukermann, E., Klavans, J. L., and Strzalkowski, T. 2003. Information retrieval. In Mitkov (2003), pp. 529–544.

Wheatley, H. B. 1878. *What is an Index? A few notes on Indexes and indexers.* London, 2nd edition.

Witten, I. H. and Frank, E. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann.