Improved Automatic Keyword Extraction Given More Linguistic Knowledge

Anette Hulth
Department of Computer and Systems Sciences
Stockholm University
Sweden
hulth@dsv.su.se

Abstract

In this paper, experiments on automatic extraction of keywords from abstracts using a supervised machine learning algorithm are discussed. The main point of this paper is that by adding linguistic knowledge to the representation (such as syntactic features), rather than relying only on statistics (such as term frequency and *n*-grams), a better result is obtained as measured by keywords previously assigned by professional indexers. In more detail, extracting NP-chunks gives a better precision than *n*-grams, and by adding the PoS tag(s) assigned to the term as a feature, a dramatic improvement of the results is obtained, independent of the term selection approach applied.

1 Introduction

Automatic keyword assignment is a research topic that has received less attention than it deserves, considering keywords' potential usefulness. Keywords may, for example, serve as a dense summary for a document, lead to improved information retrieval, or be the entrance to a document collection. However, relatively few documents have keywords assigned, and therefore finding methods to automate the assignment is desirable.

A related research area is that of terminology extraction (see e.g., Bourigault et al. (2001)), where all terms describing a domain are to be extracted. The aim of keyword assignment is to find a small set of terms that describes a specific document, independently of the domain it belongs to. However, the latter may very well benefit from the results of the former, as appropriate keywords often are of a terminological character.

In this work, the automatic keyword extraction is treated as a supervised machine learning task, an approach first proposed by Turney (2000). Two important issues are how to define the potential terms, and what features of these terms are considered discriminative, i.e., how to represent the data, and consequently what is given as input to the learning algorithm. In this paper, experiments with three term selection approaches are presented: *n*-grams; noun phrase (NP) chunks; and terms matching any of a set of part-of-speech (PoS) tag sequences. Four different features are used: term frequency, collection frequency, relative position of the first occurrence, and the PoS tag(s) assigned to the term.

2 Points of Departure

Treating the automatic keyword extraction as a supervised machine learning task means that a classifier is trained by using documents with known keywords. The 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. Turney (2000) presents results for a comparison between an extraction model based on a genetic algorithm and an implementation of bagged C4.5 decision trees for the task. The terms are all stemmed unigrams, bigrams, and trigrams from the documents, after stopword removal. The features used are, for example, the frequency of the most frequent phrase component; the relative number of characters of the phrase; the first relative occurrence of a phrase component; and whether the last word is an adjective, as judged by the unstemmed suffix. Turney reports that the genetic algorithm outputs better keywords than the decision trees. Part of the same training and test material is later used by Frank et al. (1999) for evaluating their algorithm in relation to Turney's algorithm. This algorithm, which is based on naive Bayes, uses a smaller and simpler set of features—term frequency, collection frequency (idf), and relative position—although it performs equally well. Frank et al. also discuss the addition of a fourth feature that 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.

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. The number of terms assigned must be explicitly limited by the user for these algorithms. Turney and Frank et al. report the precision for five and fifteen keywords per document. Recall is not reported in their studies. In Table 1 their results when training and testing on journal articles are shown, and the highest values for the two algorithms are presented.

	Prec.	**Corr. mean**
5 terms*	29.0	1.45
15 terms**	18.3	2.75

Table 1: Precision, and the average number of correct terms for Turney (2000)* and Frank et al. (1999)**, for five and fifteen extracted terms.

There are two drawbacks in common with the approaches proposed by Turney (2000) and Frank et al. (1999). First, the number of tokens in a keyword is limited to three. In the data used to train the classifiers evaluated in this paper, 9.1% of the manually assigned keywords consist of four tokens or more, and the longest keywords have eight tokens. Secondly, the user must state how many keywords to extract from each document, as both algorithms, for each potential keyword, output the probability of the term being a keyword. This could be solved by manually setting a threshold value for the probability, but this decision should preferably be made by the extraction system.

Finding potential terms—when no machine learning is involved in the process—by means of PoS patterns is a common approach. For example, 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 is used to determine what terms are keywords. An extraction system called *LinkIT* (see e.g., Evans et al. (2000)) compiles the phrases having a noun as the head, and then ranks these according to the heads' frequency. 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. The final example given in this paper is Daille et al. (1994) who apply statistical filters on the extracted noun phrases. In that study it is concluded that term frequency is the best filter candidate of the scores investigated. When PoS patterns are used to extract potential terms, the problem lies in how to restrict the number of terms, and only keep the ones that are relevant.

In the case of professional indexing, the terms are normally limited to a domain-specific thesaurus, but not to those present only in the document to which they are assigned. For example, Steinberger (2001) presents work where as a first step, all lemmas after stop word removal in a document are ranked according to the log-likelihood ratio, thus a list of content descriptors is obtained. These terms are then used to assign thesaurus terms, that have been automatically assigned associating lemmas during a training phase. In this paper, however, the concern is not to limit the terms to a set of allowed terms.

As opposed to Turney (2000) and Frank et al. (1999), who experiment with keyword extraction from full-length texts, this work concerns keyword extraction from abstracts. The reason for this is that many journal papers are not available as full-length texts, but as abstracts only, as is the case for example on the Internet.

The starting point for this work was to examine whether the data representation suggested by Frank et al. was adequate for constructing a keyword extraction model from and for abstracts. As the results were poor, two alternatives to extracting *n*-grams as the potential terms were explored. The first approach was to extract all noun phrases in the documents as judged by an NP-chunker. The second selection approach was to define a set of PoS tag sequences, and extract all words or sequences of words that matched any of these, relying on a PoS tagger. These two different approaches mean that the length of the potential terms is not limited to something arbitrary, but reflects a linguistic property. The solution to limiting the number of terms—as the majority of the extracted words or phrases are not keywords—was to apply a machine learning algorithm to decide which terms are keywords and which are not. The output from the machine learning algorithm is binary (a term is either a keyword or not), consequently the system itself limits the amount of extracted keywords per document. As for the features, a fourth feature was added to the ones used by Frank et al., namely the PoS tag(s) assigned to the term. This feature turned out to dramatically improve the results.

3 The Corpus

The collection used for the experiments described in this paper consists of 2 000 abstracts in English, with their corresponding title and keywords from the *Inspec* database. The abstracts are from the years 1998 to 2002, from 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, i.e., terms 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. For the experiments described here, only the uncontrolled terms were considered, as these to a larger extent are present in the abstracts (76.2% as opposed to 18.1%).

The set of abstracts was arbitrarily divided into three sets: a training set (to construct the model) consisting of 1 000 documents, a validation set (to evaluate the models, and select the best performing one) consisting of 500 documents, and a test set (to get unbiased results) with the remaining 500 abstracts. The set of manually assigned keywords were then removed from the documents. For all experiments the same training, validation, and test sets were used.

4 Building the Classifiers

This section begins with a discussion on the different ways the data were represented: in Section 4.1 the term selection approaches are described, and in Section 4.2 the features are discussed. Thereafter, a brief description of the machine learning approach is given. Finally in Section 4.4, the training and the evaluation of the classifiers are discussed.

4.1 Three Term Selection Approaches

In this section, the three different term selection approaches, in other words, the three definitions of what constitutes a term in a document, are described.

n-grams

In a first set of runs, the terms were defined in a manner similar to Turney (2000) and Frank et al. (1999). (Their studies were introduced in Section 2.) All unigrams, bigrams, and trigrams were extracted. Thereafter a stoplist was used (from Fox (1992)), where all terms beginning or ending with a stopword were removed. Finally all remaining tokens were stemmed using Porter's stemmer (Porter, 1980). In this paper, this manner of selecting terms is referred to as *the* n*-gram approach*.

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

- Only non-alphanumeric characters that were not present in any keyword in the training set were removed (keeping e.g., C++).
- Numbers were removed only if they stood separately (keeping e.g., 4YourSoul.com).

- Proper nouns were kept.
- The stemming and the stoplist applied were different.
- The stems were kept even if they appeared only once (which is true for 80.0% of the keywords present in the training set).

NP-chunks

That nouns are appropriate as content descriptors seems to be something that most agree upon. When inspecting manually assigned keywords, the vast majority turn out to be nouns or noun phrases with adjectives, and as discussed in Section 2, the research on term extraction focuses on noun patterns. To not let the selection of potential terms be an arbitrary process—which is the case when extracting *n*-grams—and better capture the idea of keywords having a certain linguistic property, I decided to experiment with noun phrases.

In the next set of experiments a partial parser[1] was used to select all NP-chunks from the documents. Experiments with both unstemmed and stemmed terms were performed. This way of defining the terms is in this paper called *the chunking approach*.

PoS Tag Patterns

As about half of the manual keywords present in the training data were lost using the chunking approach, I decided to define another term selection approach. This still captures the idea of keywords having a certain syntactic property, but is based on empirical evidence in the training data.

A set of PoS tag patterns—in total 56—were defined, and all (part-of-speech tagged) words or sequences of words that matched any of these were extracted. The patterns were those tag sequences of the manually assigned keywords, present in the training data, that occurred ten or more times. This way of defining the terms is here called *the pattern approach*. As with the chunking approach, experiments with both unstemmed and stemmed terms were performed.

Out of the 56 patterns, 51 contain one or more noun tags. To give an idea of the patterns, the five most frequently occurring ones of the keywords present in the training data are

- ADJECTIVE NOUN (singular or mass)
- NOUN NOUN (both sing. or mass)
- ADJECTIVE NOUN (plural)
- NOUN (sing. or mass) NOUN (pl.)
- NOUN (sing. or mass)

4.2 Four Features

Initially, the same features that Frank et al. (1999) used for their domain-independent experiments were used. These were

- Within-document frequency
- Collection frequency
- Relative position of the first occurrence (the proportion of the document preceding the first occurrence).

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

In addition, experiments with a fourth feature were performed. This is the PoS tag or tags assigned to the term by the same partial parser used for finding the chunks and the tag patterns. When a term consists of several tokens, the 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 the 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 Rule Induction

As usual in machine learning, the input to the learning algorithm consists of *examples*, where an example refers to the feature value vector for each, in

[1]*LT CHUNK*, available at `http://www.ltg.ed.-ac.uk/software/pos/index.html` (without the hyphen).

this case, potential keyword. An example that is a manual keyword is assigned the class *positive*, and those that are not are given the class *negative*. The machine learning approach used for the experiments is that of *rule induction*, i.e., the model that is constructed from the given examples, consists of a set of rules[2]. The strategy used to construct the rules is *recursive partitioning* (or divide-and-conquer), which has as the goal to maximise the separation between the classes for each rule.

The system used allows for different ensemble techniques to be applied, meaning that a number of classifiers are generated and then combined to predict the class. The one used for these experiments is *bagging* (Breiman, 1996). In bagging, examples 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 that then vote to classify an instance.

It should be noted that my intention is not to argue for this machine learning approach in favour of any other. However, one advantage with rules is that they may be inspected, and thus might give an insight into how the learning component makes its decisions, although this is less applicable when applying ensemble techniques.

4.4 The Training and the Evaluation

The feature values were calculated for each extracted unit in the training and the validation sets, that is for the *n*-grams, NP-chunks, stemmed NP-chunks, patterns, and the stemmed patterns respectively. In other words, the within-document frequency, the collection frequency, and the proportion of the document preceding the first appearance for each potential term were calculated. Also, the POS tag(s) for each term were extracted. In addition, as the machine learning approach is supervised, the class was added, i.e., whether the term is a manually assigned keyword or not. For the stemmed terms, a unit was considered a keyword if it was equal to a stemmed manual keyword. For the unstemmed terms, the term had to match exactly.

The measure used to evaluate the results on the validation set was the *F-score*, defined as

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

combining the *precision* and the *recall* obtained. In this study, the main concern is the precision and the recall for the examples that have been assigned the class *positive*, that is how many of the suggested keywords are correct (precision), and how many of the manually assigned keywords that are found (recall). As the proportion of correctly suggested keywords is considered equally important as the amount of terms assigned by a professional indexer that was detected, β was assigned the value 1, thus giving precision and recall equal weights.

When calculating the recall, the value for the total number of manually assigned keywords present in the documents is used, independent 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 were made for each representation, with the goal to maximise the performance as evaluated on the validation set: first the weights of the positive examples were adjusted, as the data set is unbalanced. A better performance was obtained when the positive examples in the training data outnumbered the negative ones. Thereafter experiments with bagging were performed, and also, runs with and without the POS tag feature were made. The results are presented next.

5 The Results

In this section, the results obtained by the best performing model for each approach—as judged on the validation set—when run on the previously unseen test 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 better results are possible for any of the representations.

As stemming with few exceptions led to better results on the validation set over all runs, only these values are presented in this section. In Table 2, the number of assigned terms and the number of correct terms, in total and on average per document are

[2]The system is *Rule Discovery System* from Compumine AB. `www.compumine.com`.

shown. Also, precision, recall, and the F-score are presented. For each approach, both the results with and without the PoS tag feature are given.

The length of the abstracts in the test set varies from 338 to 23 tokens (the median is 121 tokens). The number of uncontrolled terms per document is 31 to 2 (the median is 9 keywords). The total number of stemmed keywords present in the stemmed test set is 3 816, and the average number of terms is 7.63. Their distribution over the 500 documents is 27 to three documents with 0 terms, with the median being 7.

As for bagging, it was noted that although the accuracy (i.e., the number of correctly classified positive and negative examples divided by the total number of examples) improved when increasing the number of classifiers, the F-score often decreased. For the pattern approach without the tag features the best model consists of a 5-bagged classifier, for the 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 had the best performance.

5.1 Results of the *n*-gram Approach

When extracting the terms from the test set according to the *n*-gram approach, the data consisted of 42 159 negative examples, and 3 330 positive examples, thus in total 45 489 examples were classified by the trained model. Using this manner of extracting the terms meant that 12.8% of the keywords originally present in the test set were lost.

To summarise the *n*-gram approach (see Table 2), without the tag feature it finds on average 4.37 keywords per document, out of originally on average 7.63 manual keywords present in the abstracts. However, the price paid for these correct terms is high: almost 38 incorrect terms per document. When adding the fourth feature, the number of correct terms decreases slightly, while the number of incorrect terms is decreased to a third. If looking at the actual distribution of assigned terms for these two runs, this varies between 134(!) and 5 without the tag feature, and from 48 to 1 with the tag feature. The median is 40 and 14 respectively.

The F-scores ($F_{\beta=1}$) for these two runs are 17.6 and 33.9 respectively. 33.9 is the highest F-score that was achieved for the six runs presented here.

5.2 Results of the Chunking Approach

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

An F-score ($F_{\beta=1}$) of 22.7 is obtained without the PoS tag feature, and 33.0 with this feature. The number of terms on average per document is 16.38 without the tag feature, and 9.58 with it. If looking at each document, the number of keywords assigned varies from 46 to 0 (for three documents) with the median 16, and 29 to 0 (for four documents) with the median value being 9 terms.

Extracting the terms with the chunking approach meant that slightly more than half of the keywords actually present in the test set were lost, and compared to the *n*-gram approach the number of correct terms assigned was almost halved. The number of incorrect keywords, however, decreased considerably. But, the difference 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 Results of the Pattern Approach

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

The F-scores ($F_{\beta=1}$) for the two runs, displayed in Table 2, are 25.6 (without the tag feature) and 28.1 (with the tag feature). The number of terms assigned on average per document is 5.04 and 3.05 without and with the tag feature respectively. The actual number of terms assigned per document is 100 to 0 (for three documents) without the tag feature, and 46 to 0 (for four documents) with the tag feature. The median is 30 and 12 respectively.

6 Concluding Remarks and Future Work

In this paper I have shown how keyword extraction from abstracts can be achieved by using simple statistical measures as well as syntactic information from the documents, as input to a machine learning algorithm. If first considering the term selec-

Method	Assign. tot.	Assign. mean	Corr. tot.	Corr. mean	Prec.	Recall	F-score
n-gram	21 104	42.21	2 187	4.37	10.4	57.3	17.6
n-gram w. tag	7 815	15.63	1 973	3.95	25.2	51.7	**33.9**
Chunking	8 189	16.38	1 364	2.73	16.7	35.7	22.7
Chunking w. tag	4 788	9.58	1 421	2.84	**29.7**	37.2	33.0
Pattern	15 882	31.76	2 519	5.04	15.9	**66.0**	25.6
Pattern w. tag	7 012	14.02	1 523	3.05	21.7	39.9	28.1

Table 2: For each representation is shown: the number of assigned (Assign.) terms in total and mean per document; the number of correct (Corr.) terms in total and mean per document; precision; recall; and F-score. The highest value is shown in bold. The total number of manually assigned terms present in the abstracts is 3 816, and the mean is 7.63 terms per document.

tion approaches, extracting NP-chunks gives a better precision, while extracting all words or sequences of words matching any of a set of PoS tag patterns gives a higher recall compared to extracting *n*-grams. The highest F-score is obtained by one of the *n*-gram runs. The largest amount of assigned terms present in the abstracts are assigned by the pattern approach without the tag feature. The pattern approach is also the approach which keeps the largest number of assigned terms after that the data have been pre-processed. Using phrases means that the length of the potential terms is not restricted to something arbitrary, rather the terms are treat as the units they are. However, of the patterns that were selected for the experiments discussed here none was longer than four tokens. If looking at all assigned keywords in the training set, 3.0% are then ruled out as potential terms. The longest chunks in the test set that were 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 paper that this information is crucial for assigning an acceptable number of terms per document, independent of what term selection strategy is chosen.

One shortcoming of the work is that there is currently 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. In the future, the patterns should somehow be categorised reflecting their semantics, perhaps in a hierarchical manner, or morphological information could be removed.

In this paper I have not touched upon the more intricate aspects of evaluation, but simply treated the manually assigned keywords as the gold standard. This is the most severe way to evaluate a keyword extractor, as many terms might be just as good, although for one reason or another not chosen by the human indexer. Future work will examine alternative approaches to evaluation. One possibility for a more liberal evaluation could be to use human evaluators with real information needs, as done by Turney (2000). Another possibility would be to let several persons index each document, thus getting a larger set of acceptable terms to choose from. This would hopefully lead to a better precision, while recall probably would be affected negatively; the importance of recall would then need to be reconsidered.

Future work should also go in the direction of generating (as opposed to extracting) keywords, by for example exploring potential knowledge provided by a thesaurus.

Acknowledgements

For valuable comments and suggestions: Beáta Megyesi, Henrik Boström, Jussi Karlgren, Harko Verhagen, Fredrik Kilander, and the anonymous EMNLP reviewers.

References

Ken Barker and Nadia Cornacchia. 2000. Using noun phrase heads to extract document keyphrases. In *Canadian Conference on AI*.

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

Didier Bourigault, Christian Jacquemin, and Marie-Claude L'Homme, editors. 2001. *Recent Advances in Computational Terminology*. John Benjamins Publishing Company, Amsterdam.

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

Béatrice Daille, Éric Gaussier, and Jean-Marc Langé. 1994. Towards automatic extraction of monolingual and bilingual terminology. In *Proceedings of COLING-94*, pages 515–521, Kyoto, Japan.

David K. Evans, Judith L. Klavans, and Nina Wacholder. 2000. Document processing with LinkIT. In *Proceedings of the RIAO Conference*, Paris, France.

Christopher Fox. 1992. Lexical analysis and stoplists. In William B. Frakes and Ricardo Baeza-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 102–130. Prentice-Hall, New Jersey.

Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 668–673, Stockholm, Sweden.

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

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

Ralf Steinberger. 2001. Cross-lingual keyword assignment. In *Proceedings of the XVII Conference of the Spanish Society for Natural Language Processing (SEPLN'2001)*, pages 273–280, Jaén, Spain.

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