Enhancing Linguistically Oriented Automatic Keyword Extraction

Anette Hulth
Dept. of Computer and Systems Sciences
Stockholm University
SE-164 40 Kista, Sweden
hulth@dsv.su.se

Abstract

This paper presents experiments on how the performance of automatic keyword extraction can be improved, as measured by keywords previously assigned by professional indexers. The keyword extraction algorithm consists of three prediction models that are combined to decide what words or sequences of words in the documents are suitable as keywords. The models, in turn, are built using different definitions of what constitutes a term in a written document.

1 Introduction

Automatic keyword indexing is the task of finding a small set of terms that describes the content of a specific document. If the keywords are chosen from the document at hand, it is referred to as *keyword extraction*, and this is the approach taken for the work presented in this paper. Once a document has a set of keywords, they can be useful for several tasks. For example, they can be the entrance to a document collection, similar to a back-of-the-book index; they can be used to refine a query to a search engine; or they may serve as a dense summary for a specific document.

In the presented research, the decision of what words or sequences of words in the documents that are suitable as keywords are made by prediction models trained on documents with manually assigned keywords. This paper presents a number of modifications to an existing keyword extraction algorithm, as well as results of the empirical verifications.

2 Background

The approach taken to the keyword extraction task is that of supervised machine learning. This means that a set of documents with known keywords is used to train a model, which in turn is applied to select keywords to and from previously unseen documents. The keyword extraction discussed in this paper is based on work presented in Hulth (2003a) and Hulth (2003b).

In Hulth (2003a) an evaluation of three different methods to extract candidate terms from documents is presented. The methods are:

- extracting all uni-, bi, and trigrams that do not begin or end with a stopword.
- extracting all noun phrase (NP) chunks as judged by a partial parser.
- extracting all part-of-speech (PoS) tagged words or sequences of words that match any of a set of empirically defined PoS patterns.

The best performing models use four attributes. These are:

- term frequency
- collection frequency
- relative position of the first occurrence
- the POS tag or tags assigned to the term

All terms are stemmed using Porter's stemmer (Porter, 1980), and an automatically selected keyword is considered correct if it is equivalent to a stemmed manually assigned keyword. The performance of the classifiers is evaluated by calculating the F-measure for the selected keywords, with equal weight given to the precision and the recall.

In Hulth (2003b), experiments on how the performance of the keyword extraction can be improved by combining the judgement of three classifiers are presented. The classifiers differ in how the data are represented, and more specifically in how the candidate terms are selected from the documents. By only assigning keywords that are selected by at least two term selection approaches—that is by taking the majority vote—a better performance is achieved. In addition, by removing the subsumed keywords (keywords that are substrings of other selected keywords) the performance is yet higher.

The classifiers are constructed by *Rule Discovery System* (RDS), a system for rule induction[1]. This means that the models consist of rules. The applied strategy is that of *recursive partitioning*, where the resulting rules are hierarchically organised (i.e., decision trees).

The data set on which the models are trained and tested originates from the Inspec database[2], and consists of abstracts in English from scientific journal papers. The set of 2 000 documents is divided into three sets: a training set of 1 000 documents (to train the models), a validation set consisting of 500 documents (to select the best performing model, e.g., for setting the threshold value for the regression runs), and the remaining 500 documents are saved for testing (to get unbiased results). 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. For the experiments presented in this paper, only the uncontrolled terms are considered, as these to a larger extent are present in the abstracts (76.2% as opposed to 18.1% for the controlled terms). The performance is evaluated using the uncontrolled keywords as the gold standard.

In the paper, three minor improvements to the keyword extraction algorithm are presented. These concern how one of the term selection approaches extract candidate terms; how the collection frequency is calculated; and how the weights are set to the positive examples. The major focus of the paper is how the learning task is defined. For these experiments, the same machine learning system—RDS—is used as for the experiments presented by Hulth (2003a). Also the same data are used to train the models and to tune the parameters. The results of the experiments are presented in Tables 1–5, which show: the average number of keywords assigned per document (Assign.); the average number of correct keywords per document (Corr.); precision (P); recall (R); and F-measure (F). On average, 7.6 manually assigned keywords are present per document. The total number of manual keywords present in the abstracts in the test data set is 3 816, and is the number on which the recall is calculated.

[1] http://www.compumine.com

[2] http://www.iee.org/publish/inspec/

3 Refinements

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

3.1 Refining the NP-chunk Approach

It was noted in Hulth (2003b) 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 such 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 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.

	Assign.	**Corr.**	**P**	**R**	**F**
With det.	9.6	2.8	**29.7**	37.2	33.0
Without det.	15.0	4.2	27.7	**54.6**	**36.8**

Table 1: Extracting NP-chunks with and without the initial determiners *a*, *an*, and *the*.

3.2 Using a General Corpus

In the experiments presented in Hulth (2003a), only the documents present in the training, validation, and test set respectively are used for calculating the collection frequency. This means that the collection is rather homogenous. For this reason, the collection frequency is instead calculated on a set of 200 arbitrarily chosen documents from the British National Corpus (BNC). In Table 2, the performance of two runs when taking the majority vote of the three classifiers removing the subsumed terms is presented. The first run ('Abstracts') is when the collection frequency is calculated from the abstracts. The second run ('Gen. Corp.') is when the BNC documents are used for this calculation. If comparing these two runs, the F-measure increases. In other words, using a more general corpus for this calculation leads to a better performance of the automatic keyword extraction.

	Assign.	Corr.	P	R	F
Abstracts	11.1	3.8	**33.9**	49.2	40.1
Gen. Corp.	12.9	4.2	33.0	**55.6**	**41.4**

Table 2: Calculating the collection frequency from the abstracts, and from a general corpus (Gen. Corp.).

3.3 Setting the Weights

As the data set is unbalanced—there is a larger number of negative than positive examples—the positive examples are given a higher weight when training the prediction models. In the experiments discussed so far, the weights given to the positive examples are those resulting in the best performance for each individual classifier (as described in Hulth (2003a)). 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 3. As can be seen in this table, the recall decreases, while the precision and the F-measure increase.

	Assign.	Corr.	P	R	F
Individual best	12.9	4.2	33.0	**55.6**	41.4
Best combined	8.2	3.3	**40.0**	43.2	**41.6**

Table 3: Combining the classifiers with the best individual weight and with the best combination, respectively.

4 Regression vs. Classification

In the experiments presented in Hulth (2003a), 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 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 4, results for two regression runs on the test data are presented. These two runs are in Table 4 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.

	Assign.	Corr.	P	R	F
Classification	8.2	3.3	**40.0**	43.2	41.6
Regression	10.8	4.2	38.9	**54.8**	**45.5**
Reg. ind. thresh.	11.3	4.2	37.1	54.7	44.2

Table 4: Using classification and regression. 'Reg. ind. thesh.' refers to a run where the regression value from each model contributes only if it is over a certain threshold.

4.1 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 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 threshold are considered, and the candidate terms with the highest values are selected first. The best performance is when the 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 5 ('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 5: 3–12).

	Assign.	**Corr.**	**P**	**R**	**F**
All	10.8	4.2	38.9	**54.8**	**45.5**
Max. 12	8.6	3.6	**41.6**	46.8	44.0
3–12	8.6	3.6	41.5	46.9	44.0

Table 5: Assigning all terms over the threshold (All), and limiting the number of terms assigned per document (Max. 12, and 3–12 respectively).

5 Concluding Remarks

In this paper, a number of experiments leading to a better performance of a keyword extraction algorithm has 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 collection frequency is calculated, where the F-measure of the extraction 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".

Evaluating automatically extracted keywords is not trivial, as different persons may prefer different terms at different occasions. This is also true for professional indexers, where the consistency also depends on how experienced an indexer is. For example, Bureau van Dijk (1995) has shown that the index consistency between experienced indexers may be up to 60–80 per cent, while it is not unusual that it is as low as 20–30 between inexperienced indexers. The approach taken to the evaluation of the experiments presented in this paper is that of using keywords previously assigned by professional indexers as a gold standard for calculating the precision, the recall, and the F-measure. 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%. Thus the performance of the keyword extraction algorithm is at least as consistent as that of inexperienced professional indexers. 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[3].

Acknowledgements

Thanks to Henrik Boström for valuable discussions and comments.

References

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. 1984. *Classification and Regression Trees*. Chapman & Hall, New York.

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

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

Anette Hulth. 2003b. Reducing false positives by expert combination in automatic keyword indexing. In *Proceedings of the Conference on Recent Advances in Natural Language Processing (RANLP 2003)*, pages 197–203, Borovets, Bulgaria.

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

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