Reducing Over-generation Errors for Automatic Keyphrase Extraction using Integer Linear Programming

Florian Boudin

Reducing Over-generation Errors for Automatic Keyphrase Extraction using Integer Linear Programming

Florian Boudin
LINA - UMR CNRS 6241, Université de Nantes, France
florian.boudin@univ-nantes.fr

Abstract

We introduce a global inference model for keyphrase extraction that reduces over-generation errors by weighting sets of keyphrase candidates according to their component words. Our model can be applied on top of any supervised or unsupervised word weighting function. Experimental results show a substantial improvement over commonly used word-based ranking approaches.

1 Introduction

Keyphrases are words or phrases that capture the main topics discussed in a document. Automatically extracted keyphrases have been found to be useful for many natural language processing and information retrieval tasks, such as summarization (Litvak and Last, 2008), opinion mining (Berend, 2011) or text categorization (Hulth and Megyesi, 2006). Despite considerable research effort, the automatic extraction of keyphrases that match those of human experts remains challenging (Kim et al., 2010).

Recent work has shown that most errors made by state-of-the-art keyphrase extraction systems are due to over-generation (Hasan and Ng, 2014). Over-generation errors occur when a system correctly outputs a keyphrase because it contains an important word, but at the same time erroneously predicts other keyphrase candidates as keyphrases because they contain the same word. One reason these errors are frequent is that many unsupervised systems rank candidates according to the weights of their component words, e.g. (Wan and Xiao, 2008a; Liu et al., 2009), and many supervised systems use unigrams as features, e.g. (Turney, 2000; Nguyen and Luong, 2010).

While weighting words instead of phrases may seem rather blunt, it offers several advantages. In practice, words are usually much easier to extract, match and weight, especially for short documents where many phrases may not be statistically frequent (Liu et al., 2011).

Selecting keyphrase candidates according to their component words may also turn out to be useful for reducing over-generation errors if one can ensure that the importance of each word is counted only once in the set of extracted keyphrases. To do so, keyphrases should be extracted as a set rather than independently. Finding the optimal set of keyphrases is a combinatorial optimisation problem, and can be formulated as an integer linear program (ILP) which can be solved exactly using off-the-shelf solvers.

In this work, we propose an ILP formulation for keyphrase extraction that can be applied on top of any word weighting scheme. Through experiments carried out on the SemEval dataset (Kim et al., 2010), we show that our model increases the performance of both supervised and unsupervised word weighting keyphrase extraction methods.

The rest of this paper is organized as follows. In Section 2, we describe our ILP model for keyphrase extraction. Our experiments are presented in Section 3. In Section 4, we briefly review the previous work, and we conclude in Section 5.

2 Method

Our global inference model for keyphrase extraction consists of three steps. First, keyphrase candidates are extracted from the document using heuristic rules. Second, words are weighted using either supervised or unsupervised methods. Third, finding the optimal subset of keyphrase candidates is cast as an ILP and solved using an off-the-shelf solver.

2.1 Keyphrase candidate selection

Candidate selection is the task of identifying the words or phrases that have properties similar to

Proceedings of the ACL 2015 Workshop on Novel Computational Approaches to Keyphrase Extraction, pages 19–24,
Beijing, China, July 30, 2015.

those of manually assigned keyphrases. First, we apply the following pre-processing steps to the document: sentence segmentation[1], word tokenization[2] and Part-Of-Speech (POS) tagging[3].

Following previous work (Wan and Xiao, 2008a; Bougouin et al., 2013), we use the sequences of nouns and adjectives as keyphrase candidates. Candidates that have less than three characters, that contain only adjectives, or that contain stop-words[4] are filtered out. These heuristic rules are designed to avoid spurious instances and keep the number of candidates to a minimum (Hasan and Ng, 2014). All words are stemmed using Porter's stemmer (Porter, 1980).

2.2 Word weighting functions

The performance of our model depends on how word weights are estimated. Here, we experiment with three methods for assigning importance weights to words. The first two are unsupervised weighting functions, namely TF×IDF (Spärck Jones, 1972) and TextRank (Mihalcea and Tarau, 2004), which have been extensively used in prior work (Hasan and Ng, 2010). We also apply a supervised model for predicting word importance based on (Hong and Nenkova, 2014).

2.2.1 TF×IDF

The weight of each word t is estimated using its frequency $tf(t,d)$ in the document d and how many other documents include t (inverse document frequency), and is defined as:

$$\text{TF} \times \text{IDF}(t,d) = tf(t,d) \times log(D/D_t)$$

where D is the total number of documents and D_t is the number of documents containing t.

2.2.2 TextRank

A co-occurrence graph is first built from the document in which nodes are words and edges represent the number of times two words co-occur in the same sentence. TextRank (Mihalcea and Tarau, 2004), a graph-based ranking algorithm, is then used to compute the importance weight of each word. Let d be a damping factor[5], the TextRank score $S(V_i)$ of a node V_i is initialized to a default value and computed iteratively until convergence using the following equation:

$$S(V_i) = (1-d) + \left(d \times \sum_{V_j \in \mathcal{N}(V_i)} \frac{w_{ji} \times S(V_j)}{\sum_{V_k \in \mathcal{N}(V_j)} w_{jk}} \right)$$

where $\mathcal{N}(V_i)$ is the set of nodes connected to V_i and w_{ji} is the weight of the edge between nodes V_j and V_i.

TextRank implements the concept of "voting", i.e. a word is important if it is highly connected to other words and if it is connected to important words.

2.2.3 Logistic regression

We train a logistic regression model[6] for assigning importance weights to words in the document based on (Hong and Nenkova, 2014). Reference keyphrases in the training data are used to generate positive and negative examples. For a word in the document (restricted to adjectives and nouns), we assign label 1 if the word appears in the corresponding reference keyphrases, otherwise we assign 0. We use the relative position of the first occurrence, the presence in the first sentence and the TF×IDF weight as features. These features have been extensively used in supervised keyphrase extraction approaches, and have been shown to perform consistently well (Hasan and Ng, 2014).

2.3 ILP model definition

Our model is an adaptation of the concept-based ILP model for summarization introduced by (Gillick and Favre, 2009), in which sentence selection is cast as an instance of the budgeted maximum coverage problem[7]. The key assumption of our model is that the value of a set of keyphrase candidates is defined as the sum of the weights of the unique words it contains. That way, a set of candidates only benefits from including each word once. Words are thus assumed to be independent, that is, the value of including a word is not affected by the presence of any other word in the set of keyphrases.

Formally, let w_i be the weight of word i, x_i and c_j two binary variables indicating the pres-

[1] We use Punkt Sentence Tokenizer from NLTK.

[2] We use Penn Treebank Tokenizer from NLTK.

[3] We use the Stanford Part-Of-Speech Tagger (Toutanova et al., 2003).

[4] We use the english stop-list from NLTK.

[5] We set d to 0.85 as in (Mihalcea and Tarau, 2004).

[6] We use the Logistic Regression classifier from scikit-learn with default parameters.

[7] Given a collection S of sets with associated costs and a budget L, find a subset $S' \subseteq S$ such that the total cost of sets in S' does not exceed L, and the total weight of elements covered by S' is maximized (Khuller et al., 1999).

ence of word i and candidate j in the set of extracted keyphrases, Occ_{ij} an indicator of the occurrence of word i in candidate j and N the maximum number of extracted keyphrases, our model is described as:

$$\max \quad \sum_i w_i x_i \tag{1}$$

$$s.t. \quad \sum_j c_j \leq N \tag{2}$$

$$c_j Occ_{ij} \leq x_i, \quad \forall i, j \tag{3}$$

$$\sum_j c_j Occ_{ij} \geq x_i, \quad \forall i \tag{4}$$

$$x_i \in \{0, 1\} \quad \forall i$$

$$c_j \in \{0, 1\} \quad \forall j$$

The constraints formalized in equations 3 and 4 ensure the consistency of the solution: selecting a candidate leads to the selection of all the words it contains, and selecting a word is only possible if it is present in at least one selected candidate.

By summing over word weights, this model overly favors long candidates. Indeed, given two keyphrase candidates, one being included in the other (e.g. *uddi registries* and *multiple uddi registries*), this model always selects the longest one as its contribution to the objective function is larger. To correct this bias, a regularization term is added to the objective function:

$$\max \quad \sum_i w_i x_i - \lambda \sum_j \frac{(l_j - 1)c_j}{1 + substr_j} \tag{5}$$

where l_j is the size, in words, of candidate j, and $substr_j$ the number of times c_j occurs as a subtring in the other candidates. This regularization penalizes the candidates that are composed of more than two words, and is dampened for candidates that occur frequently as substrings in other candidates. Here, we assume that for multiple candidates of the same size, the one that is less frequent in the document should be stressed first.

The resulting ILP is then solved exactly using an off-the-shelf solver[8]. The solving process takes less than a second per document on average. The N candidate keyphrases returned by the solver are selected as keyphrases.

[8]We use GLPK, `http://www.gnu.org/software/glpk/`

3 Experiments

3.1 Experimental settings

We carry out our experiments on the SemEval dataset (Kim et al., 2010), which is composed of scientific articles collected from the ACM Digital Library. The dataset is divided into training (144 documents) and test (100 documents) sets. We use the set of combined author- and reader-assigned keyphrases as reference keyphrases.

We follow the common practice (Kim et al., 2010) and evaluate the performance of our method in terms of precision (P), recall (R) and f-measure (F) at the top N keyphrases[9]. Extracted and reference keyphrases are stemmed to reduce the number of mismatches.

For each word weighting function, namely TF×IDF, TextRank and Logistic regression, we compare the performance of our ILP model (hereafter `ilp`) with that of two word-based weighting baselines. The first baseline (hereafter `sum`) simply ranks keyphrase candidates according to the sum of the weights of their component words as in (Wan and Xiao, 2008b; Wan and Xiao, 2008a). The second baseline (hereafter `norm`) consists in scoring keyphrase candidates by computing the sum of the weights of their component words normalized by their length as in (Boudin, 2013).

As a post-processing step, we remove redundant keyphrases from the ranked lists generated by both baselines. A keyphrase is considered redundant if it is included in another keyphrase that is ranked higher in the list.

IDF weights are computed on the training set. The regularization parameter λ is set, for all the experiments, to the value that achieves the best performance on the training set, that is 0.3 for TF×IDF, 0.4 for TextRank and 1.2 for Logistic regression.

3.2 Results

The performance of our model on top of different word weighting functions is shown in Table 1. Overall, our model consistently improves the performance over the baselines. We observe that the results for `sum` are very low. Summing the word weights favors long candidates and is prone to over-generation errors, as illustrated by the example in Table 2.

[9]Scores are computed using the evaluation script provided by the SemEval organizers.

Weighting + Ranking	Top-5 candidates			Top-10 candidates		
	P	R	F	P	R	F
TF×IDF + `sum`	5.6	1.9	2.8	5.3	3.5	4.2
+ `norm`	19.2	6.7	9.9	15.1	10.6	12.3
+ `ilp`	25.4	9.1	13.3†	17.5	12.4	14.4†
TextRank + `sum`	4.5	1.6	2.3	4.0	2.8	3.3
+ `norm`	18.8	6.6	9.6	14.5	10.1	11.8
+ `ilp`	22.6	8.0	11.7†	17.4	12.2	14.2†
Logistic regression + `sum`	4.2	1.5	2.2	4.7	3.4	3.9
+ `norm`	23.8	8.3	12.2	18.9	13.3	15.5
+ `ilp`	29.4	10.4	15.3†	19.8	14.1	16.3

Table 1: Comparison of TF×IDF, TextRank and Logistic regression for different ranking strategies when extracting a maximum of 5 and 10 keyphrases. Results are expressed as a percentage of precision (P), recall (R) and f-measure (F). † indicates significance at the 0.05 level using Student's t-test.

Normalizing the candidate scores by their lengths (`norm`) produces shorter candidates but does not limit the number of over-generation errors. As we can see from the example in Table 2, 9 out of 10 extracted keyphrases are containing the word *nugget*. Our ILP model removes these redundant keyphrases by controlling the impact of each word on the set of extracted keyphrases. The resulting set of keyphrases is more diverse and thus increases the coverage of the topics addressed in the document.

Note that the reported results are not on par with keyphrase extraction systems that use ad-hoc pre-processing, involve structural features and leverage external resources. Rather our goal in this work is to demonstrate a simple and intuitive model for reducing over-generation errors.

TF×IDF + `sum` (P = 0.1)
advertis bid; certain advertis budget; keyword bid; convex hull landscap; budget optim bid; **uniform bid strategi**; advertis slot; advertis campaign; ward advertis; searchbas advertis
TF×IDF + `norm` (P = 0.2)
advertis; advertis bid; **keyword**; keyword bid; landscap; advertis slot; advertis campaign; ward advertis; searchbas advertis; advertis random
TF×IDF + `ilp` (P = 0.4)
click; **advertis**; uniform bid; landscap; **auction**; convex hull; **keyword**; **budget optim**; single-bid strategi; queri

Table 2: Example of the top-10 extracted keyphrases for the document J-3 of the SemEval dataset. Keyphrases are stemmed and whose that match reference keyphrases are marked bold.

4 Related Work

In recent years, keyphrase extraction has attracted considerable attention and many different approaches were proposed. Generally speaking, keyphrase extraction methods can be divided into two main categories: supervised and unsupervised approaches.

Supervised approaches treat keyphrase extraction as a binary classification task, where each phrase is labeled as keyphrase or non-keyphrase (Witten et al., 1999; Turney, 2000; Kim and Kan, 2009; Lopez and Romary, 2010). Unsupervised approaches usually rank phrases by importance and select the top-ranked ones as keyphrases. Methods for ranking phrases include graph-based ranking (Mihalcea and Tarau, 2004; Wan and Xiao, 2008a; Wan and Xiao, 2008b; Bougouin et al., 2013; Boudin, 2013), topic-based clustering (Liu et al., 2009; Liu et al., 2010; Bougouin et al., 2013), statistical models (Paukkeri and Honkela, 2010; El-Beltagy and Rafea, 2010) and language modeling (Tomokiyo and Hurst, 2003).

The work of (Ding et al., 2011) is perhaps the closest to our present work. They proposed an ILP formulation of the keyphrase extraction prob-

lem that combines TF×IDF and position features in an objective function subject to constraints of coherence and coverage. In their model, coherence is measured by Mutual Information and coverage is estimated using Latent Dirichlet Allocation (LDA) (Blei et al., 2003). Their work differs from ours in that (1) it is phrased-based and thus does not penalize redundant keyphrases, and (2) it requires estimating a large number of hyperparameters which makes it difficult to generalize.

5 Conclusion and Future Work

In this paper, we proposed an ILP formulation for keyphrase extraction that reduces over-generation errors by weighting keyphrase candidates as a set rather than independently. In our model, keyphrases are selected according to their component words, and the weight of each unique word is counted only once. Experiments show a substantial improvement over commonly used word-based ranking approaches using either supervised and unsupervised weighting schemes.

In future work, we intend to extend our model to include word relatedness through the use of association measures. By doing so, we expect to better differentiate semantically related keyphrase candidates according to the association strength between their component words.

Acknowledgments

This work was partially supported by the GOLEM project (grant of CNRS PEPS FaSciDo 2015, `http://boudinfl.github.io/GOLEM/`). We thank the anonymous reviewers, Adrien Bougouin and Evgeny Gurevsky for their insightful comments.

References

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March.

Florian Boudin. 2013. A comparison of centrality measures for graph-based keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 834–838, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Zhuoye Ding, Qi Zhang, and Xuanjing Huang. 2011. Keyphrase extraction from online news using binary integer programming. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 165–173, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Samhaa R. El-Beltagy and Ahmed Rafea. 2010. Kp-miner: Participation in semeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 190–193, Uppsala, Sweden, July. Association for Computational Linguistics.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*, pages 10–18, Boulder, Colorado, June. Association for Computational Linguistics.

Kazi Saidul Hasan and Vincent Ng. 2010. Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In *Coling 2010: Posters*, pages 365–373, Beijing, China, August. Coling 2010 Organizing Committee.

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

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 712–721, Gothenburg, Sweden, April. Association for Computational Linguistics.

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

Samir Khuller, Anna Moss, and Joseph (Seffi) Naor. 1999. The budgeted maximum coverage problem. *Information Processing Letters*, 70(1):39 – 45.

Su Nam Kim and Min-Yen Kan. 2009. Re-examining automatic keyphrase extraction approaches in scientific articles. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications*, pages 9–16, Singapore, August. Association for Computational Linguistics.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010. Semeval-2010 task 5 : Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26, Uppsala, Sweden, July. Association for Computational Linguistics.

Marina Litvak and Mark Last. 2008. Graph-based keyword extraction for single-document summarization. In *Coling 2008: Proceedings of the workshop Multi-source Multilingual Information Extraction and Summarization*, pages 17–24, Manchester, UK, August. Coling 2008 Organizing Committee.

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 257–266, Singapore, August. Association for Computational Linguistics.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 366–376, Cambridge, MA, October. Association for Computational Linguistics.

Zhiyuan Liu, Xinxiong Chen, Yabin Zheng, and Maosong Sun. 2011. Automatic keyphrase extraction by bridging vocabulary gap. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 135–144, Portland, Oregon, USA, June. Association for Computational Linguistics.

Patrice Lopez and Laurent Romary. 2010. Humb: Automatic key term extraction from scientific articles in grobid. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 248–251, Uppsala, Sweden, July. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 404–411, Barcelona, Spain, July. Association for Computational Linguistics.

Thuy Dung Nguyen and Minh-Thang Luong. 2010. Wingnus: Keyphrase extraction utilizing document logical structure. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 166–169, Uppsala, Sweden, July. Association for Computational Linguistics.

Mari-Sanna Paukkeri and Timo Honkela. 2010. Likey: Unsupervised language-independent keyphrase extraction. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 162–165, Uppsala, Sweden, July. Association for Computational Linguistics.

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

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

Takashi Tomokiyo and Matthew Hurst. 2003. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment - Volume 18*, MWE '03, pages 33–40, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technology - Volume 1 (NAACL)*, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

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

Xiaojun Wan and Jianguo Xiao. 2008b. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2*, AAAI'08, pages 855–860. AAAI Press.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Kea: Practical automatic keyphrase extraction. In *Proceedings of the Fourth ACM Conference on Digital Libraries*, DL '99, pages 254–255, New York, NY, USA. ACM.