PKU_ICL at SemEval-2017 Task 10: Keyphrase Extraction with Model Ensemble and External Knowledge

Liang Wang
Key Laboratory of
Computational Linguistics
Peking University, China
intfloat@pku.edu.cn

Sujian Li
Key Laboratory of
Computational Linguistics
Peking University, China
lisujian@pku.edu.cn

Abstract

This paper presents a system that participated in *SemEval 2017 Task 10* (subtask A and subtask B): *Extracting Keyphrases and Relations from Scientific Publications* (Augenstein et al., 2017). Our proposed approach utilizes external knowledge to enrich feature representation of candidate keyphrase, including Wikipedia, IEEE taxonomy and pre-trained word embeddings etc. Ensemble of unsupervised models, random forest and linear models are used for candidate keyphrase ranking and keyphrase type classification. Our system achieves the 3rd place in subtask A and 4th place in subtask B.

1 Introduction

Keyphrases summarize the most important aspects of a document. They can be helpful in many areas such as information retrieval, topic modeling and text classification. However, manually labeling keyphrase would be far too time-consuming, and unrealistic especially when dealing with web-scale collection of documents. Therefore, automatic keyphrase extraction has drawn growing interests among NLP research communities for years.

For state-of-the-art system on keyphrase extraction, Hasan and Ng(2014) presents a comprehensive survey. Their experiments demonstrate that unsupervised approaches including graph-based ranking and topic modeling techniques perform best on *News* and *Blogs* dataset. In *SemEval 2010 Task 5* (Kim et al., 2010) (Kim et al., 2013), which also aims to tackle the challenge of keyphrase extraction in scientific area, a majority of the participants adopt supervised approaches, and especially the top 2 systems are both supervised. Thus, in our work, we argue that supervised approaches can enable combination of rich features, with parameters learned efficiently and automatically, while their unsupervised counterparts often involve simply designed features and manually fine-tuned hyperparameters.

Based on the consideration above, for *SemEval 2017 Task 10*, our system is designed as a supervised one which also explore unsupervised techniques as auxiliaries. It involves three steps: *candidate generation*, *keyphrase ranking* and *keyphrase type classification*. For *candidate generation*, we use chunking-based approach to discover phrases that match a predefined part-of-speech pattern. Heuristic rules are manually designed by experience and applied to filter out those phrases which are unlikely to be keyphrases. For *keyphrase ranking* in subtask A, we use a straightforward regression-based pointwise ranking method. Here, two unsupervised algorithms *TextRank* (Mihalcea and Tarau, 2004) and *SGRank* (Danesh et al., 2015) are incorporated into random forest by providing their output as complementary features. In our experiments, we find that stacking linear model upon random forest can provide extra performance gain. For *keyphrase type classification* in subtask B, we model it as a three-way classification problem, with the same feature set and classifiers used in subtask A.

Feature engineering is a critical part for supervised model. The task of keyphrase extraction heavily relies on statistical features(such as TF-IDF) and semantic features. However, due to the limited size of labeled dataset, it is hard to get reliable estimation of phrases' IDF value or semantic representation. In this paper, we solve this problem by exploiting external knowledge resources such as *Wikipedia* and pre-trained word embeddings. Experiments show the effectiveness of our proposed feature set.

2 Methodology

Our system works in a pipeline fashion. It involves *candidate generation*, *keyphrase ranking* for subtask A and *keyphrase type classification* for subtask B. As the third step use the same feature set and classifiers as second step, we omit its detailed description.

2.1 Keyphrase Candidate Generation

There are generally two approaches for candidate generation: n-grams and part-of-speech pattern matching. Even though n-grams strategy usually achieves higher recall, it also brings in more false positives, which could cause serious problem for classifiers. Our strategy is a combination of part-of-speech pattern matching and heuristic rules.

First, let's define the heuristic rules with the format of functions, which take a phrase p as an argument and output a boolean value. Assume $NP = (NN * |JJ*) * (NN*)$.

$$f_1(p) = \text{whether } p \text{ matches pattern } NP \tag{1}$$
$$f_2(p) = \text{whether } p \text{ consists of capital letters} \tag{2}$$
$$f_3(p) = \text{whether } p \text{ consists of <5 words} \tag{3}$$
$$f_4(p) = \text{whether } p \text{ contains `[' or `]' or `,'} \tag{4}$$
$$f_5(p) = \text{whether } p \text{ consists of only digits} \tag{5}$$

A phrase p becomes a keyphrase candidate if $(f_1(p) \vee f_2(p)) \wedge f_3(p) \wedge \neg f_4(p) \wedge \neg f_5(p)$ is $true$. Candidate generation is carried out at sentence level, we don't consider the possibility that a keyphrase may span across multiple sentences.

2.2 External Knowledge

To deal with the aforementioned problem, we exploit several external knowledge resources to get more reliable estimation of statistical and semantic features.

- **English Wikipedia.**[1] It consists of more than 5 million articles covering almost every area you can think of. We use this corpus D to calculate IDF of word t. Words with top $10k$ IDF score are kept.

$$IDF(t, D) = log\frac{|D|}{|\{d \in D : t \in d\}|} \tag{6}$$

- **IEEE taxonomy.** Official *IEEE taxonomy*[2] includes a list of manually summarized keyphrases related to technical areas. Articles in this shared task come from three domains: computer science, material science and physics. All three domains are covered by IEEE. We add a boolean feature which indicates whether the given candidate keyphrase appears in this list.

- **Pre-trained Glove embeddings.**[3] (Pennington et al., 2014) Word embeddings trained on billions of tokens provide a simple way to incorporate semantic knowledge. They prove to be helpful in many NLP tasks especially when labeled data is limited. In our system, we use IDF-weighted word embeddings for phrase representation. Given a phrase consisting of n words $w_1, w_2...w_n$, its representation is calculated as follows.

$$\frac{\sum_{i=1}^{n} IDF(w_i) \cdot E_{w_i}}{\sum_{i=1}^{n} IDF(w_i)} \tag{7}$$

E_{w_i} is the glove embedding of word w_i.

2.3 Feature Engineering

Based upon the experience of many previous work on keyphrase extraction and the unique characteristics of scientific publications, our system incorporates four types of features: linguistic features, context features, external knowledge based features and unsupervised model based features, as shown in Table 4.

feature type	feature definition
linguistic features	stemmed unigram
	avg/max/min of TF/IDF/TF·IDF
	whether p consists of capital letters
	part-of-speech for every word in p
	number of tokens/characters in p
context features	previous/next token of p
	POS of previous/next token of p
	distance between p and citation
	relative position of p in given text
external knowledge	whether p is in *IEEE taxonomy* list
	Wikipedia based avg/max/min IDF
	glove embedding of p
unsupervised feature	whether p is in top 20 keyphrases according to *TextRank* algorithm
	whether p is in top 20 keyphrases according to *SGRank* algorithm

Table 1: Features for a candidate keyphrase p.

2.4 Model Ensemble for Keyphrase Ranking

Model ensemble has been shown to be an effective way to boost generalization performance both

[1]https://dumps.wikimedia.org/enwiki/

[2]https://www.ieee.org/documents/taxonomy_v101.pdf

[3]http://nlp.stanford.edu/projects/glove/

in practice and theoretically (Zhou, 2012). Random forest itself is an ensemble model, with variant of decision trees combined via *Bagging*. In this shared task, we explore two layers of stacking.

At the first layer, we stack trees upon output from unsupervised algorithms. There are numerous algorithms for unsupervised keyphrase extraction based on clustering, graph-based ranking etc, different algorithms reflect different aspects of phrase. Stacking provides a convenient and powerful way to combine those information. In this paper, we use two algorithms: *TextRank* (Mihalcea and Tarau, 2004) and *SGRank* (Danesh et al., 2015).

At the second layer, we stack linear model upon random forest. Instead of treating decision tree as a classifier, it can be seen as learning to transform input features. Each leaf node corresponds to a feature transformation path from root node, and therefore can serve as a boolean feature. Linear model can be applied to learn the weights of those features. Logistic regression is usually used, however, we find linear SVM is more robust to overfitting in this shared task.

For keyphrase ranking in subtask A, probabilistic score is required, candidates with score no less than α are chosen as keyphrases. α is tuned on validation dataset to balance precision and recall. For keyphrase type classification in subtask B, it is a three-way classification problem.

In deep learning community, "stacking" usually means joint training of multiple layers. In this paper, "stacking" means that lower layers provide their output as features for upper layer, different layers are trained separately.

3 Experiments

For details about this shared task and dataset, please refer to *SemEval 2017 Task 10* description paper (Augenstein et al., 2017).

3.1 Experimental Setup

Preprocessing We use *nltk* (Bird, 2006) to segment each paragraph into a list of sentences, tokenize every sentence and then get part-of-speech tag for every token. *Snowball Stemmer* is used for stemming. Stop words, punctuations and digits are removed for feature engineering, but not for keyphrase candidate generation. We use simple heuristics to parse the IEEE taxonomy pdf file and get 6978 phrases in total.

Configurations Library *scikit-learn* is used for implementation of our supervised models. Random forest is set to have 200 trees and other parameters are set to default. Parameters of linear SVM are all set to default. We use 50-dimensional glove embeddings for calculating phrase representation. For subtask A, we choose threshold $\alpha = 0.15$ to balance recall and precision.

3.2 Results and Analysis

		precision	recall	f1-score
subtask A		0.522	0.498	**0.510**
subtask B	Material	0.464	0.456	0.460
	Process	0.441	0.364	0.399
	Task	0.286	0.041	0.072
	Average	0.450	0.374	**0.409**

Table 2: Official results on test set.

Our system's final results are shown in Table 2, f1-score for subtask A is 0.510 (3rd place), and micro-averaged f1-score for subtask B is 0.409 (4th place). The f1-score of the 1st place solution in a similar task *SemEval 2010 Task 5* is 27.5% (Kim et al., 2010). In comparison with the prior work, our system seems to be surprisingly well. We attribute such performance gap to unique characteristics of this shared task. Instead of keyphrase extraction from entire document, participants are only asked to extract keyphrase from single paragraph, and the density of keyphrases is higher.

Another interesting phenomenon is the poor numbers for *Task* keyphrases. Most of *Material* and *Process* keyphrases are noun phrases or have capital letters, they are relatively easy to discriminate by part-of-speech pattern. However, *Task* keyphrases cover a wide range of part-of-speech patterns, and some of them have verb or conjunction. It remains a challenge for our system to achieve satisfying performance for *Task* keyphrases.

	Material	Process	Task	Micro-average
recall	0.715	0.608	0.334	0.606

Table 3: Recall for keyphrases.

An important metric for our pipeline system is recall for keyphrases in candidate generation step. Table 3 shows that our heuristic rules cover 60.6% of keyphrases in training data, although it's possible to improve recall by introducing more part-of-

	subtask A			subtask B		
	precision	recall	f1-score	precision	recall	f1-score
unsupervised features	0.378	0.341	0.359	0.186	0.168	0.177
+ linguistic features	0.481	0.417	0.447	0.352	0.305	0.327
+ context features	0.499	0.497	0.498	0.371	0.369	0.370
+ external knowledge	**0.522**	**0.498**	**0.510**	**0.450**	**0.374**	**0.409**

Table 4: Performance with different combinations of features.

speech patterns, precision will go lower. There has to be a tradeoff between recall and precision.

	precision	recall	f1-score
tf-idf	0.163	0.216	0.186
rf	0.510	0.507	0.508
rf+svm	**0.524**	0.520	0.522
best	0.510	**0.610**	**0.560**

Table 5: Comparison of different models on test data. (a) *tf-idf* output phrases with top 15 tf·idf score; (b) *rf* stands for random forest; (c) *rf+svm* stacks a linear SVM upon random forest; (d) *best* is the 1st place solution for this shared task.

Table 5 shows the effectiveness of our approach. Even though *rf* and *rf+svm* share the same input features and random forest already has a built-in ensemble mechanism, *rf+svm* model still manages to improve all three metrics via stacking, with f1-score increasing by 1.4%, from 50.8% to 52.2%.

We also examine how different feature combinations would affect overall performance. Results are shown in Table 4. Unsupervised features are pretty impressive to discriminate keyphrase and non-keyphrase (subtask A), but they fail to reliably identify keyphrase type (subtask B). Incorporation of external knowledge is clearly a key to further boost system performance. All six metrics improve as more features are added. Once again it shows our model's ability to combine many features without worrying too much about overfitting. It is possible to add more relevant features.

4 Conclusion

This paper gives a brief description of our system at *SemEval 2017 Task 10* for keyphrase extraction of scientific papers. By incorporating multiple external knowledge resources, careful feature engineering and model ensemble, our system achieves competitive performance.

Acknowledgements

We would like to thank *SemEval 2017* organizers and anonymous reviewers for their helpful suggestions. This work was partially supported by National Natural Science Foundation of China (61572049 and 61333018). The correspondence author of this paper is Sujian Li.

References

Isabelle Augenstein, Mrinal Kanti Das, Sebastian Riedel, Lakshmi Nair Vikraman, and Andrew McCallum. 2017. SemEval 2017 Task 10: ScienceIE - Extracting Keyphrases and Relations from Scientific Publications. In *Proceedings of the International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Vancouver, Canada.

Steven Bird. 2006. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics, pages 69–72.

Soheil Danesh, Tamara Sumner, and James H Martin. 2015. Sgrank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. *Lexical and Computational Semantics* page 117.

Kazi Saidul Hasan and Vincent Ng. 2014. Automatic keyphrase extraction: A survey of the state of the art. In *ACL (1)*. pages 1262–1273.

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*. Association for Computational Linguistics, pages 21–26.

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

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Zhi-Hua Zhou. 2012. *Ensemble methods: foundations and algorithms*. CRC press.