Comparing Different Keywords Extration Methods

Wuhao Wang

December 15, 2022

Abstract

Keyword extraction is a important task in NLP fields, many advanced tasks are based on keyword extraction. In this project, TF-IDF, Keybert, TextRank and Yake will be compared. The dataset is from other NLP competations and the evaluation is based on f1-score. We found that Keybert and Yake are with the highest f1-score while TF-IDF is the most efficient method. Textrank is not flexible since it can not customize the number of keywords.

1 Introduction

Keywords detecing is always a important task in many text applications. In this project, four methods will be discussed and compared. The first methods is TF-IDF[1], which basically calculate and compare the frequency of a specific word in all the text data and one text data. Higher TF-IDF value means higher importance of this word to the text. Another model is known as Yake[4][2][3]. Yake can be condidered as an extension of TF-IDF, this algorithm will calculate TF-IDF value and also cosine similarity, then a word graph will be constructed to obtain the weights of each candidate word. The third one is a self-supervised learning method called KeyBert [7]. In this approach, document-level vector representations are extracted by BERT. Subsequently, word vectors are extracted for the N-gram words/phrases, and then, cosine similarity will be applied to find the most similar word/phrase to the document. Finally, the selected words will be identified as the words that best describe the entire document. Last Algorithm is TextRank[9]. The TextRank algorithm is a graph-based sorting algorithm. The work flows are as follows:putting text segmentation into several constituent units (e.g. sentences), building a node connection graph, using similarity between sentences as the edge Weights, calculate the TextRank value of the sentence through loop iterations, and finally extract the sentences with high rankings and combine them into text summaries.

[1] https://en.wikipedia.org/wiki/Tf/$%$E2/$%$80$%$93idf

2 Related work

In 2018, Bijoyan Das and Sarit Chakraborty raised the method of using TF-IDF to grab keywords that can describe sentiment in the text [5]. Shahzad Qaiser and Ramsha Ali used TF-IDF to find the words with highest relevance to the documents [10]. Alex Graves used LSTM and other network to perform keyword extraction which gave the inspiration of Keybert[6].

3 Data Set

The data set used in this project is Inspec [8]. The language is English, the original text is collected from scientific paper within computer science fields.Below is the numeric features of the data.

N-docs	total Goldkey	Tokens per doc	Absent Goldkey
2000	29230	128.2	37.7

N-docs is the number of documents contained in the data, **total Goldkey** is the number of gold keywords in the dataset, **tokens per doc** is average number of tokens in each document, **Absent Godkey** is the percentage of gold keywords that absent in the documents.

4 Evaluation

In the competation [1]hold by Association for Computational Linguistics in Vancouver, Canada, they use f1-score to evaluate the results. In this project, all the model performance will be evaluated by f1-score in this project. The f1-score can be calculated by:

$$\begin{aligned} f1 &= \frac{2}{recall^{-}1 + precision^{-}1} \\ &= 2\,\frac{precision \times recall}{precision + recall} \\ &= \frac{2tp}{2tp + fp + fn} \end{aligned}$$

In this project, **true positive** means if a gold keyword is predicted as a gold keyword, **false negative** means if a gold keyword is predicted as a non gold keyword,**false positive** means if a non gold keyword is predicted as a gold keyword.

5 Experiment

In this section, the results of each method will be presented and discussed.

5.1 TF-IDF

For TF-IDF method, this project is referring to spacy package[2]. The results of TF-IDF method are as follows. N is number of keywords that model would extract, f1 is the corresponding f1-score.

N	10	12	16	18	20
f1	0.226	0.242	0.281	0.266	0.239

5.2 Yake

For yake model, this project is referring to official package[3]. The results of yake are as follows. N is number of keywords that model would extract, f1 is the corresponding f1-score. Note that there are many other parameters can be tuned, see appendix for more details.

N	10	12	16	18	20
f1	0.291	0.327	0.381	0.367	0.345

5.3 Keybert

For Keybert model, this project is referring to official package[4]. The results of Keybert are as follows. N is number of keywords that model would extract, f1 is the corresponding f1-score. Note that there are many other parameters can be tuned, see appendix for more details.

N	10	12	16	18	20
f1	0.268	0.284	0.350	0.301	0.238

5.4 Textrank

For textrank model, this project is referring to two different implementations,one is summaNLP[5], another one is spacy [6]. Textrank algorithm does not offer "number of keywords" parameter, the results of two different implementations are as follows:

implementations	summaNLP	Spacy
f1	0.200	0.179

[2] https://github.com/explosion/spaCy
[3] https://github.com/LIAAD/yake
[4] https://github.com/MaartenGr/KeyBERT
[5] https://github.com/summanlp/textrank
[6] https://spacy.io/universe/project/spacy-pytextrank

6 Conclusion

From the experiment results, one can easily see that yake and keybert are much better than other two methods. For this dataset, every document contains averagely 15 golden keywords. And for textrank which does not receive "number of keywords" parameter, the f1-score is lower because of too few keywords prediction.In this project, We also try to set "number of keywords = 5" for other models and results are quite similar to textrank.In one word,TF-IDF methods is the fastest method among these four methods while keybert and yake have the highest f1-score. Note that in Inspec

dataset, 37.7% of keywords do not show up in the text, which means the 'fn' in the f1-score is atleast 0.37. That is a important reason why all the models can not have high f1-score. The drawback of TF-IDF is obvious.

TF-IDF methodcan only get every single keyword. When it comes to key phrase extraction(KPE), TF-IDF method can not work. In this project, the dataset semeval2017 [1] ,which contains key phrase,is also applied to all of these four models. We pick the same hyperparameters as in the Inspec datasetm, but the length of key phrase is set in a range from 1 to 4.

models	TF-IDF	Keybert	TextRank	Yake
f1	0.04	0.13	0.12	0.15

7 Discussion

Yake is similar to TF-IDF, the weights of each word comes from document-level statistics. But Yake considers more features than TF-IDF and the f1-score also reflects that Yake is more advanced than TF-IDF. Keybert

uses BERT to extract the word vectors and considers similarity between words and documents. So, the result is highly depends on which pre-trained network is used. Textrank is based on graph algorithm. Like transformer-

based model, it will divide text into different part. The different thing is that Textrank will consider those elements as points and edges in a graph. Then a voting system will select the most important words.

8 Future Work

Based on this project, there are multiple future tasks. The most important thing is thinking how can TF-IDF method predict key phrase? Maybe this

can be achieved by combining the candidate keywords and re-calculate the weights. Another thinking is how to improve other models' f1-score. In the

semeval-2017 competation, the highest team reach the f1-score of 0.68. It is a quite good results but is still far away from the practical usage standard.

9 Appendix

A Hyperparameter

For Keybert

hp	diversity	use_maxsum	model
value	0.2	False	'all-MiniLM-L6-v2'

For Yake

hp	deduplication_threshold	deduplication_algo	windowSize
value	0.9	'seqm'	1

References

[1] Isabelle Augenstein, Mrinal Das, Sebastian Riedel, Lakshmi Vikraman, and Andrew McCallum. SemEval 2017 task 10: ScienceIE - extracting keyphrases and relations from scientific publications. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 546–555, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[2] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. Yake! keyword extraction from single documents using multiple local features. *Inf. Sci.*, 509(C):257–289, jan 2020.

[3] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. Yake! collection-independent automatic keyword extractor. In Gabriella Pasi, Benjamin Piwowarski, Leif Azzopardi, and Allan Hanbury, editors, *Advances in Information Retrieval*, pages 806–810, Cham, 2018. Springer International Publishing.

[4] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289, 2020.

[5] Bijoyan Das and Sarit Chakraborty. An improved text sentiment classification model using TF-IDF and next word negation. *CoRR*, abs/1806.06407, 2018.

[6] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602–610, 2005. IJCNN 2005.

[7] Maarten Grootendorst. Keybert: Minimal keyword extraction with bert., 2020.

[8] Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, page 216–223, USA, 2003. Association for Computational Linguistics.

[9] Rada Mihalcea and Paul Tarau. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[10] Shahzad Qaiser and Ramsha Ali. Text mining: Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181, 07 2018.