Incorporating Corpus Term Frequency into TF.IDF for Imrpoved Keyphrase Extraction Accuracy

Place licence statement here for the camera-ready version, see Section “Licence Statement” of the instructions for preparing a manuscript (coling2014.pdf).

|  |  |  |
| --- | --- | --- |
| [name]  [address]  [address]  [address]  [e-mail] | [name]  [address]  [address]  [address]  [e-mail] | [name]  [address]  [address]  [address]  [e-mail] |

|  |  |
| --- | --- |
|  |  |

Abstract

Keyphrase extraction is one of the fundamental tasks in natural language processing. Currently term-frequency inverse-document-frequency (TF.IDF) serves as a competitive baseline for unsupervised keyphrase extraction and a fundamental feature for more sophisticated supervised and unsupervised keyphrase extraction algorithms. It is one of the most widely used methods in practice due to its accuracy, computational efficiency and ease of implementation.

TF.IDF uses two statistical features of a term: the number of times it occurs in one document and the portion of documents in the corpus which contain it. We augment the TF.IDF product by incorporating the number of times a term occurs in all documents in the corpus and show that our proposed term weighting function outperforms TF.IDF in keyphrase extraction accuracy in multiple datasets.

1. Introduction

Keyphrase extraction is the process of distinguishing terms in a text that best signify its meaning and the topics it is about. Designing accurate and efficient keyphrase extraction algorithms is one of the most important tasks in natural language processing. Keyphrase extraction is used in semantic tagging [15], document clustering [4,22] and classification [8]. In content-based recommender systems, extracted keywords are used to generate personalized recommendations [14]. In advertisement keyphrases are used to take into account the ad’s context [24]. Keyphrases extracted from bioinformatics texts are used to build gene interaction networks [13]. Keyphrase extraction has also been used for facilitating contextual search [2] and building ontologies [3]. These diverse uses indicate keyphrase extraction as an integral part of many tools and algorithms that we have constructed for managing the vast, and ever increasing, amount of information shared as unstructured text.

One of the most enduring and still widely used techniques for keyphrase extraction is ranking terms by their TF.IDF score and selecting top ranked terms as keyphrases. In General TF.IDF is one of the most widely used term weighting schemes [18]. In keyphrase extraction it remains a competitive and robust baseline [12, 5]. In the 2010 Semeval keyphrase extraction competition TF.IDF was used as the unsupervised baseline where it outperformed the two other supervised baselines [11].

Not only is TF.IDF widely used on its own it is also a main feature of many more sophisticated algorithms. Two salient examples of this can again be found in the Semeval 2010 competition. The best performing system named HUMB [25], which utilizes a supervised approach, uses the TF.IDF score of a term to quantify its informativness, which is one of the three “Content features” used and therefore presumably makes a considerable contribution to overall performance. TF.IDF is also central to KP-Miner [1] which was the best performing unsupervised keyphrase extraction algorithm in the Semeval 2010 competition. The KP-miner algorithm has three stages: (1) Keyphrase candidate selection, (2) assigning weights to candidates and (3) refining the weight-based ranking using additional features. The weight assignment function used in step 2 is TF.IDF.B where B is a boosting factor proportional to the number of words in the candidate keyphrase.

As the above examples show TF.IDF plays a fundamental role in both supervised and unsupervised state-of-the-art algorithms for keyphrase extraction. Therefore an improvement on TF.IDF could potentially improve both supervised and unsupervised state-of-the-art performances.

In the current work we propose just such an improvement to TF.IDF: Our augmented version of TF.IDF outperforms the original in both precision and recall on four datasets while conserving its computational simplicity. Therefore it is a suitable candidate for replacing TF.IDF in more sophisticated algorithms

In the following sections we will provide an overview of our approach, discuss similar and related work and finally compare the performance of our proposed weighting measure against TF.IDF on various corpuses.

1. Approach

TF.IDF has many variations but in its simplest form it can be described as:

Where is the number of occurrences of the term in document is the total number of documents in the corpus and is the number of documents that contain . As seen above, TF.IDF only considers the frequency of the term in the document in which it occurs. Our intuition is to extend the term frequency heuristic to documents beyond the one containing the term. The basic intuition is that if a term is frequent not only in the document at hand but also in other documents in the corpus then it is even more likely to be a keyphrase. A simple statistic that can be used to represent a term’s frequency in all documents in which it occurs is average term frequency, , as defined in equation 1 below. To incorporate this measure into TF.IDF we simply add it as a third term to the TF.IDF product producing the function which we name :

Our initial experiments showed that while indeed outperforms TF.IDF on some datasets, on other datasets it is outperformed by TF.IDF. We hypothesize that this is due to the existence of terms in these datasets that are not keyphrases but have a large and are therefore ranked highly by causing it to have a lower accuracy. To better explain this intuition we use the following hypothetical example: Imagine in a large corpus of computer science abstracts the term ‘Graph’ has a corpus term frequency of 1000 and a document frequency of 100. Thereforeμfor graph is 10. However graph itself is not a keyphrase in any of the documents it occurs in but in many cases it is a part of multiword keyphrases such as ‘Graph traversal’ and ‘Graph theory’. Let us further assume, in this hypothetical example, that ‘Graph theory’ has a corpus term frequency of 200 and a document frequency of 20 and ‘Graph traversal’ has a corpus term frequency of 100 and a document frequency of 10. Notice that all three terms, graph, graph theory and graph traversal have the same whereas our aim is to assign higher scores to keyphrases, graph theory and graph traversal, and a lower weight to non-keyphrases such as graph. However in this example does not distinuigsh much rarer keyphrases from a non-keyphrase. To mitigate the effect of non-keyphrases with high s in the term weighting scheme we use the log function to transform making it more sensitive to the rareness of a term. Through experimentation we came up with two modifications of as described in equations (3) and (4)

In the previous example graph, graph theory and graph traversal all have the same . However both and assign higher weights to less frequent terms. Therefore ,in this case, the largest values for both and belong to graph traversal followed by graph theory. The non-keyphrase, graph, is ranked lowest by both which is more inline with our overall purpose of assigning higher weights to keyphrases.

Table [1] summarizes the above example and shows the values assigned to the different terms by , and . As can be seen from this table and values are larger for rarer terms where as assigns the same weight to all three terms.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | ctf | df |  |  |  |
| Graph | 1000 | 100 | 10 | 0.03 | 1.5 |
| Graph theory | 200 | 20 | 10 | 0.11 | 1.7 |
| Graph traversal | 100 | 10 | 10 | 0.2 | 2.0 |

Table 1. Hypotheitcal corpus statistics and corresponding for three terms

Similar to how is produced by multiplying and TF.IDF other weighting functions, and are produced using and respectively.

and are able to outperform TF.IDF on datasets where cannot. However this comes at the cost of being outperformed by TF.IDF on some datasets where outperforms. Given these three weighing methods, , , and each of which outperforms TF.IDF on some datasets while none outperforming it on all datasets we construct a combined weighting method by calculating their product as seen in equation 7. In other words the final term weight is the geometric mean of , , and but without calculating the cube root of the product, as taking the root would affect all terms equally and would not add to the distinguishing power of the term weighting function.

Expressed in terms of corpus statistics:

Notice that the only piece of information needed for calculating , in addition to and terms, which are necessary for calculating TF.IDF, is the corpus term frequency (). Given , can be computed in constant time. Therefore our proposed term weighting method is on the same order of space and time complexity as TF.IDF.

From a practical point of view, systems that use TF.IDF weighting often construct a lookup table where the corpus frequency of a term can be looked up in constant time. The corpus frequency of all terms can be calculated with a single pass through the corpus. For a system to use the corpus term frequency of all terms can be calculated in the same pass through the corpus. The term’s can then be stored along with its by allowing one additional space in the lookup table.

In this work similar to the Semeval 2010 TF.IDF baseline [11] we use an n-gram based approach where all uni-, bi- and tri-grams in the document are considered as keyphrase candidate terms.

1. Related Works

One of the oldest and still widely used term weighting methods is TF.IDF. The abbreviation TF stands for term frequency. Using term frequency as a heuristic for determining important words in the text was first suggested by Salton and Lesk [20] in 1968. It was based on the natural intuition that a word which occurs more times in a document is more likely to be a keyphrase. IDF which stands for inverse document frequency was proposed by Jones in [7] as a method of incorporating the rareness of a word in estimating its importance. The basic intuition behind IDF is that words which occur in more documents in a given corpus are less likely to be important to the meaning of any specific one. Therefore IDF assigns higher weights to terms that occur in fewer documents. In their 1975 paper Slaton et al. put term frequency and inverse document frequency together as TF.IDF [19]. The intuition being that TF would distinguish relevant terms while IDF would weed out unimportant ones. In [18] Robertson provides an overview of the different theoretical explanations of IDF.

Although TF.IDF was first introduced in the context of information retrieval it has been used as and remains a competitive baseline for unsupervised keyword extraction [12, 5].

In [21] Tian et al suggest an improvement to TF.IDF by incorporating not only the frequency of a term but also its distribution. They argue that the occurrences of an important term are more likely to be evenly spread out in the document whereas terms that occur frequently only in certain parts of the document are more likely to be important to those parts but not the document as a whole. The authors further argue that the opposite is true about the term’s distribution in the corpus: Terms that are less uniformly distributed in the corpus are more likely to be of importance to their respective documents. The idea that a less uniform corpus distribution corresponds to a higher likelihood of a term being a keyphrase is very similar to our intuition that could be used as a heuristic to better distinguish keyphrases.

To the best of our knowledge our approach for incorporating corpus term frequency into the TF.IDF product to improve keyphrase extraction is novel. In [26] average term frequency () is referred to as density and used to construct a weighting scheme for keyphrase extraction named TF.Density which is proportional to a term’s , as is our method. However their weighting function is very different from ours. Furthermore their method is specialized for news corpuses and takes into account temporal information regarding a document, as it would represent a news event, whereas our weighting scheme is more general purpose.

In the context of information retrieval has been suggested for assigning weights to terms in short queries [10] as an alternative to TF.IDF. More recently in 2013 an enhancement to TF.IDF term weighting for query-based retrieval has been proposed in [17]. Similar to our approach, in this work IDF is multiplied by a function of . However the term added to the TF.IDF product is , which is different from our incorporation of as seen in equation 8. Furthermore we have focused on keyphrase extraction and show noticeable improvement whereas the aim in [17] is to improve query based retrieval accuracy and they report a small contribution to their performance from incorporating in the aforementioned manner.

1. Evaluaition

We compare our proposed term weighting schemes () with TF.IDF on four datasets. The first is the Semeval 2010 keyphrase extraction dataset which consists of 284 full-length articles from the ACM digital library. Each article has two sets of keywords assigned to it, one set is assigned by the author, the other by a designated group of readers. In our evaluation, similar to the Semeval task, we use the union of the sets of keyphrases assigned by readers and the author to calculate precision and recall. A more detailed description of this dataset and the Semeval task is available in [11]. We compare term rankings produced using against those produced by TF.IDF using the precision and recall at k metric. That is, we compute the precision and recall for the k top ranked terms for each algorithm, for different k. The precision recall curves that will follow in this section are for k from 1 to 30.

In our experiments any 1, 2 and 3-gram was considered a term and a keyphrase candidate. No stop word filtering was performed. This is because some keyphrases contain stop words such as ‘Of’. All words in the corpus were stemmed. Figure 1 shows the precision recall curves for the Semeval corpus. Table 1 shows the F1 scores for the Semeval corpus and the improvements made by over TF.IDF at k = 5,10 and 15.

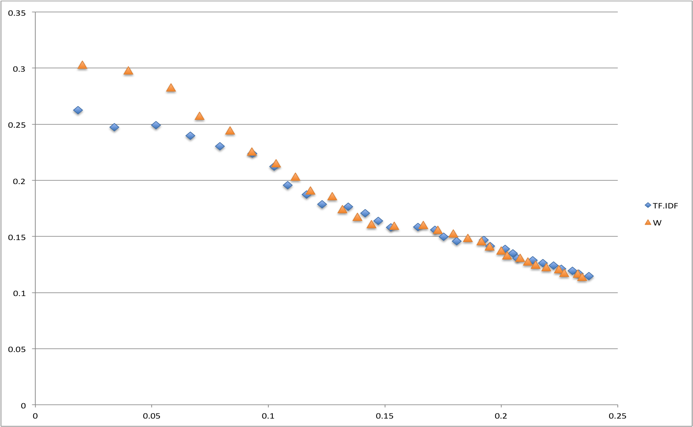


Figure 1. Precision Recall and F-measure improvements for Semeval corpus

|  |  |  |  |
| --- | --- | --- | --- |
| K= | TF.IDF | W | Improvement |
| 5 | 0.117 | 0.124 | 5.64% |
| 10 | 0.145 | 0.151 | 3.73% |
| 15 | 0.161 | 0.163 | 0.33% |

The second dataset we use in our evaluation is a set of 1094 ACM article abstracts. Each abstract has a number of keywords assigned which we use as the gold standard. This dataset has been used in [23], which contains a more detailed description of it, for similar experiments. Figure 2 shows the precision recall curves on the ACM dataset. Table 2 provides F1 scores and improvement percentages for the ACM dataset.

|  |  |  |  |
| --- | --- | --- | --- |
| K= | TF.IDF | W | Improvement |
| 5 | 0.106 | 0.116 | 9.37 % |
| 10 | 0.092 | 0.103 | 11.9% |
| 15 | 0.077 | 0.089 | 15.1% |

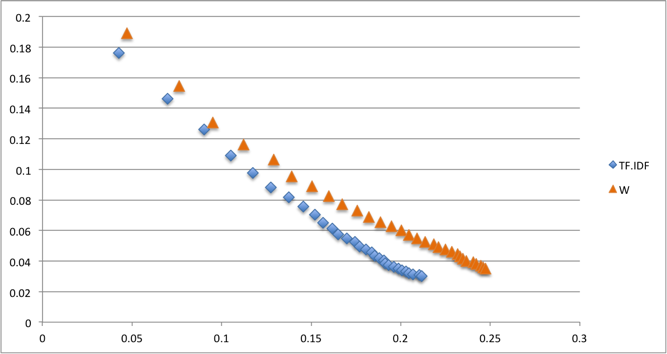


Figure 2. Precision Recall and F-measure improvements for ACM corpus

The third dataset used in our evaluation is the Inspeq corpus consisting of 2000 abstracts from journal articles in computer science, control and information sciences. This dataset has been used for keyphrase extraction experiments by Hulth [6] and Mihalcea [16]. Similar to [6] and [16] we use the uncontrolled human assigned keyphrases as the golden set and test on 500 abstracts in the test set. Figure 3 shows the precision recall curves for the Inspeq dataset. Table 3 provides the F1 scores and improvements made by .

|  |  |  |  |
| --- | --- | --- | --- |
| K= | TF.IDF | W | improvement |
| 5 | 0.114 | 0.123 | 7.89% |
| 10 | 0.126 | 0.138 | 9.52% |
| 15 | 0.123 | 0.135 | 9.65% |

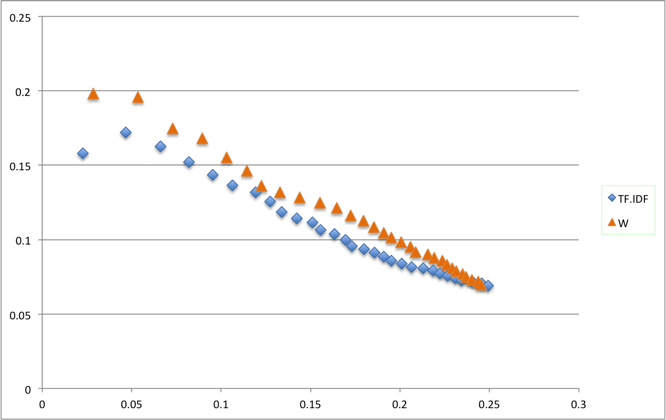


Figure 3. Precision Recall and F-measure improvements for Inspeq corpus

Finally the fourth dataset that we have tested on is a corpus of 32604 short news snippets (Downloaded from http://acube.di.unipi.it/tmn-dataset/ ) from mainstream news outlets such as New York Times, Fortune magazine, etc. [27] These snippets cover all areas of news such as sports, politics, entertainments, etc. The average length of a snippet is 185 words. No keyphrases are assigned to the snippets in the original dataset. We had two regular readers of news manually annotate 113 snippets. The readers were instructed to choose phrases from a snippet that best signify what it is about. Disagreements on keyphrases were resolved by discussion. The average number of keyphrases per snippet is 2.5 with a maximum of 5 and a minimum of 1.The average length of a keyphrase is 1.5 words. Figure 4 shows the precision recall curves for this news dataset. Table 4 shows the F1 scores for TF.IDF, and improvement percentages.

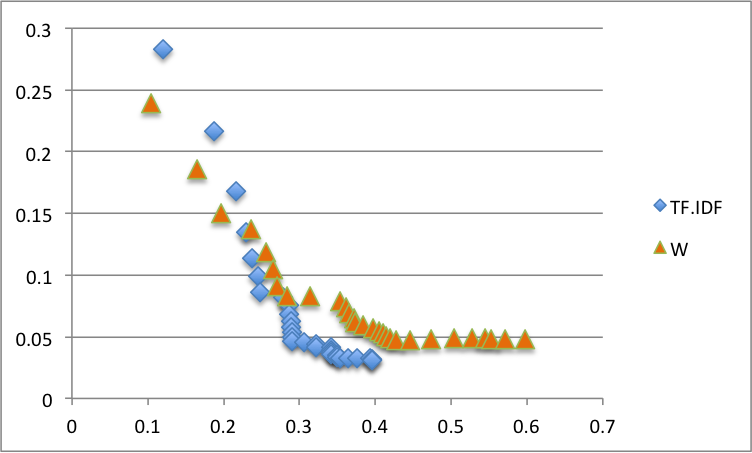


Figure 4. Precision Recall and F-measure improvements for news corpus

|  |  |  |  |
| --- | --- | --- | --- |
| K= | TF.IDF | W | Improvement |
| 5 | 0.153 | 0.162 | 5.6 % |
| 10 | 0.11 | 0.171 | 17.1% |
| 15 | 0.08 | 0.105 | 27.6% |

1. Discussion

As can be seen W outperforms TF.IDF in all cases. The datasets we have tested on include one containing a relatively small number of large documents (Semeval corpus), two datasets containing a relatively large number of small documents (ACM and Inspeq corpuses) and finally the news dataset which contains a very large number of very short text snippets. The fact that our proposed measure outperforms TF.IDF on all of these datasets is an indication of its robustness and indicates the possibility that it might be suitable as a general-purpose enhancement to TF.IDF. We should note that the Semeval, ACM and Inspeq datasets contain some human assigned keyphrases that are not found in the text of the document but are external topics deemed relevant by human annotators. This causes the maximum possible precision for keyphrase extraction to be less than 100%. However because both and TF.IDF are subject to the same evaluation procedure this shortcoming of the dataset does not hinder comparison.

Among the four corpuses the smallest gains are seen in the Semeval dataset. We hypothesize this to be due to the small number of documents in the Semeval corpus compared to the other three. Semeval contains only 284 documents whereas the other corpuses have more than a thousand each. The small number of documents in the Semeval corpus makes it less likely for several documents to be about the same topic. This in turn makes it less likely for several documents to contain the same phrase as a keyphrase. This reduces the effectiveness of in distinguishing keyphrases because it relies on keyphrases that are common to several documents, and occur in them with high frequency, thereby having a relatively high corpus term frequency to document frequency ratio. Conversely the largest gains on the F1 measure are seen on the news dataset. This, we hypothesize, is because the news dataset is more likely to contain documents that share a topic due to the large number of documents in this corpus and the a news event is usually covered by multiple news sources. As a side note, we acknowledge the smallness of the news corpus’ test set. We plan on expanding the test set in future works.

As mentioned, our final weighing scheme is the result of combining three simpler functions and as seen in equations 4, 5, 6 and 7. While outperforms TF.IDF on all datasets, it is itself outperformed by at least one of its constituent coefficients on each dataset. For example figure 5 below shows the precision recall curves for TF.IDF, and and on the Inspeq dataset where outperforms both other measures.

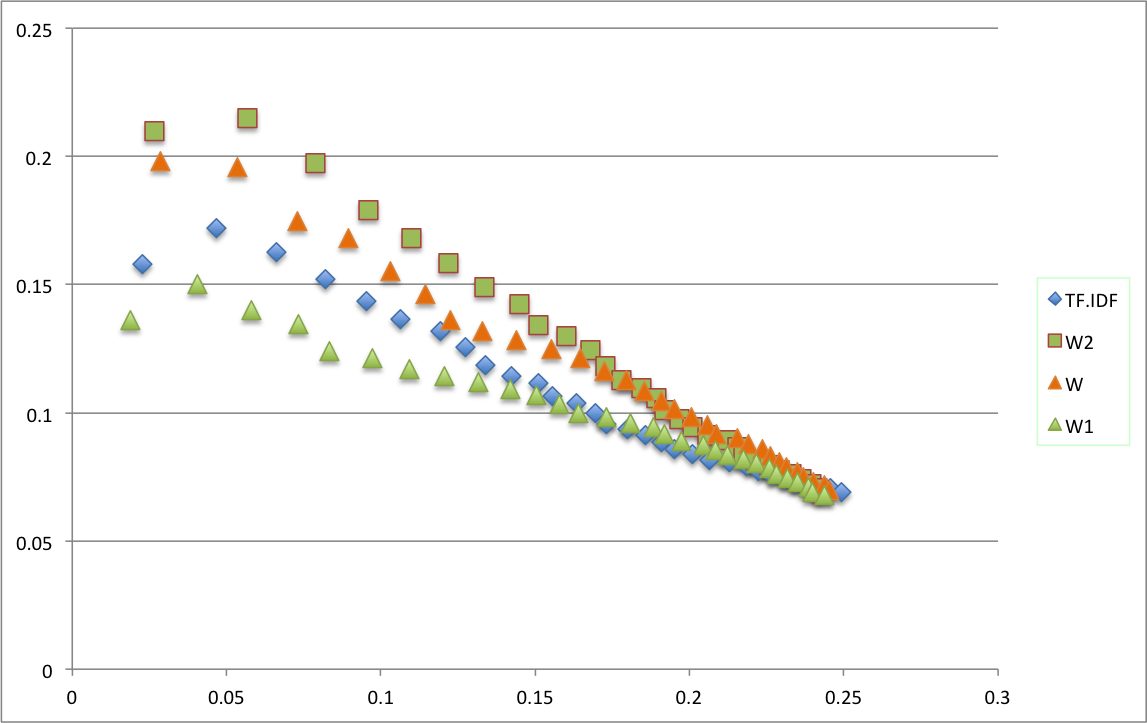


Figure 5. Precision recall curves for , , and TF.IDF on the Inspeq corp.

Therefore an approach where the best performing function amongst and different combinations of its constituent coefficients is chosen for a corpus based on its performance on a corresponding training set also seems plausible.

Figure 5 also shows outperforming on the Inspeq dataset. The reason outperforms on this dataset could be due to the existence of non-keyphrase terms with high s. In the hypothetical example presnted in the approach section we showed how is insensitive to frequncy whereas assigns higher weights to rarer terms. As a result , which is a function of , as seen in equation 2, is outperformed by which is a function of as in equation 5. One conveivable group of high-frequency non-keyphrase terms in a dataset could be comprised of words that are not keyphrases themselves but are part of multi-word keyphrases. This is similar to the scenario presented in our hypothetical example in the approach section. Therefore it could be hypothesized that the keyphrases for such a dataset are more likely to be multiwords compared to a dataset where outperforms . This is indeed what we see with the ACM and Inspeq datasets. As mentioned, figure 5 depicts outperforming on the Inspeq dataset. Similartly figure 6 shows outperforming on the ACM dataset.

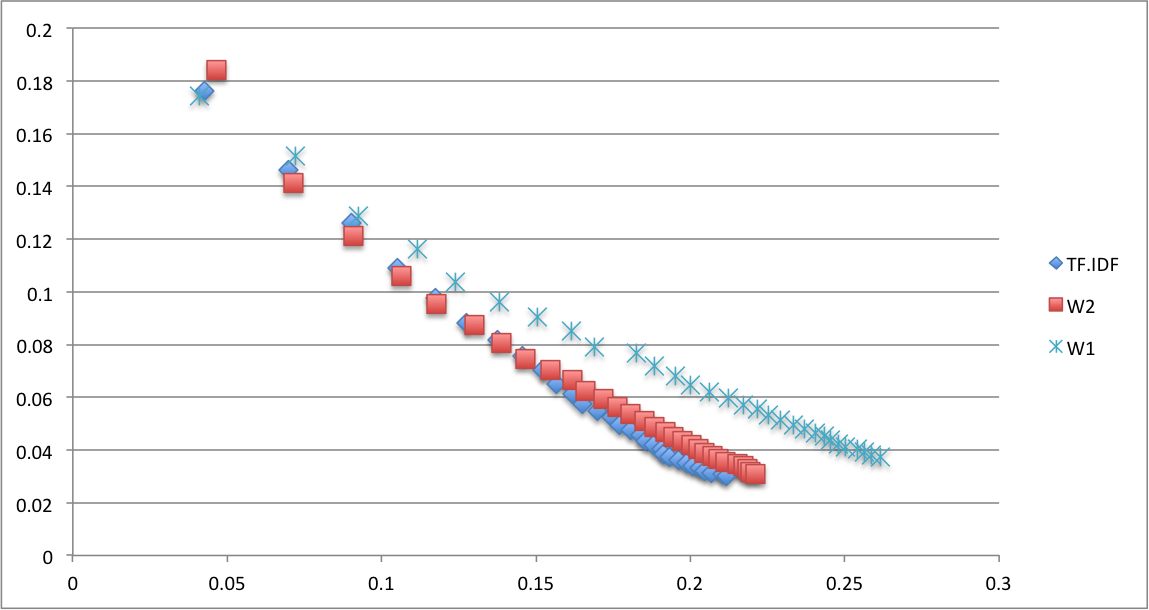


Figure 6. Precision recall curves for , and TF.IDF on the ACM corp.

The average number of words in a keyphrase in the Inspeq dataset is 2.32 but 1.87 in the ACM dataset. The fact that the Inspeq dataset has generally longer keyphrases along with the higher performance of on this dataset may indeed serve as an indication of the liklihood of scenarios similar to the hypothetical example presented in the approach seciton.

Conclusion

TF.IDF term ranking is a common technique for distinguishing important terms in text. Furthermore it is a core feature in many more sophisticated keyphrase extraction algorithms, notably the best performing supervised and unsupervised systems in the Semeval 2010 keyphrase extraction shared task. In this work we have proposed an improvement to TF.IDF that incorporates corpus term frequency. An important future direction for our research would be to replace TF.IDF with our proposed measure in state-of-the-art algorithms to see whether the improvements we see over TF.IDF propagate to the final output in these complex algorithms. Other tasks such as information retrieval and document classification that use TF.IDF to build vector space representations of documents may also see improvements by using our proposed method.

Reference

1. El-Beltagy, S. R., & Rafea, A. 2010 . Kp-miner: Participation in semeval-2. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 190-193). Association for Computational Linguistics.
2. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppin, E. 2001. Placing search in context: The concept revisited. In*Proceedings of the 10th international conference on World Wide Web* (pp. 406-414). ACM.
3. Fortuna, B., Mladenič, D., & Grobelnik, M. 2006 . *Semi-automatic construction of topic ontologies* (pp. 121-131). Springer Berlin Heidelberg.
4. Hammouda, K. M., Matute, D. N., & Kamel, M. S. 2005. Corephrase: Keyphrase extraction for document clustering. In *Machine Learning and Data Mining in Pattern Recognition* (pp. 265-274). Springer Berlin Heidelberg.
5. Hasan, K. S., & Ng, V. 2010. Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (pp. 365-373). Association for Computational Linguistics.
6. Hulth, A. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing* (pp. 216-223). Association for Computational Linguistics.
7. Jones, K. "A statistical interpretation of term specificity and its application in retrieval." *Journal of documentation* 28.1 (1972): 11-21.
8. Karanikolas, N. N., & Skourlas, C. 2004. Key-phrase extraction for classification. In MEDICON and HEALTH TELEMATICS 2004: X Mediterranean Conference on Medical and Biological Engineering.
9. Kim, S., et al. "Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles." *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2010.
10. Kwok, K. "A new method of weighting query terms for ad-hoc retrieval." Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1996.
11. Kim, S. N., Medelyan, O., Kan, M. Y., & Baldwin, T. 2010. Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In*Proceedings of the 5th International Workshop on Semantic Evaluation* (pp. 21-26). Association for Computational Linguistics.
12. Liu, F., Pennell, D., Liu, F., & Liu, Y. 2009. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics* (pp. 620-628). Association for Computational Linguistics.
13. Liu, Y., Ciliax, B. J., Borges, K., Dasigi, V., Ram, A., Navathe, S. B., & Dingledine, R. 2004. Comparison of two schemes for automatic keyword extraction from MEDLINE for functional gene clustering. In *Computational Systems Bioinformatics Conference, 2004. CSB 2004. Proceedings. 2004 IEEE* (pp. 394-404). IEEE.
14. Lops, P., de Gemmis, M., & Semeraro, G. 2011. Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook* (pp. 73-105). Springer US.
15. Medelyan, O., Frank, E., & Witten, I. H. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3* (pp. 1318-1327). Association for Computational Linguistics.
16. Mihalcea R., and Paul T. "TextRank: Bringing order into texts."*Proceedings of EMNLP*. Vol. 4. No. 4. 2004.
17. Paik, J. "A novel TF-IDF weighting scheme for effective ranking."*Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2013.
18. Robertson, S. 2004. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of documentation*, *60*(5), 503-520.
19. Salton, G., Anita W. and Chung-Shu Y. "A vector space model for automatic indexing." *Communications of the ACM* 18.11 (1975): 613-620.
20. Salton, G., & Lesk, M. E. 1968. Computer evaluation of indexing and text processing. *Journal of the ACM (JACM)*, *15*(1), 8-36.
21. Tian, X. and Wang T. "An improvement to TF: Term Distribution based Term Weight Algorithm." *Networks Security Wireless Communications and Trusted Computing (NSWCTC)*
22. Tonella, P., Ricca, F., Pianta, E., & Girardi, C. 2003. Using keyword extraction for web site clustering. In *Web Site Evolution, 2003. Theme: Architecture. Proceedings. Fifth IEEE International Workshop on* (pp. 41-48). IEEE.
23. Wartena, C., Rogier B. and Wout S. "Keyword extraction using word co-occurrence." *Database and Expert Systems Applications (DEXA), 2010 Workshop on*. IEEE, 2010.
24. Wu, X., & Bolivar, A. 2008. Keyword extraction for contextual advertisement. In *Proceedings of the 17th international conference on World Wide Web* (pp. 1195-1196). ACM
25. Lopez, P., & Romary, L. 2010 . HUMB: Automatic key term extraction from scientific articles in GROBID. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 248-251). Association for Computational Linguistics.
26. Liu, S. W., & Chang, H. T. 2013. A topic detection and tracking system with TF-Density. In *Recent Progress in Data Engineering and Internet Technology*(pp. 115-120). Springer Berlin Heidelberg.
27. Phan, X. H., Nguyen, L. M., & Horiguchi, S. (2008, April). Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on World Wide Web* (pp. 91-100). ACM.