International Journal of Computer Applications (0975 – 8887)
Volume 181 – No.1, July 2018

Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents

Shahzad Qaiser
School of Computing
Universiti Utara Malaysia
Sintok, Kedah, Malaysia

Ramsha Ali
School of Quantitative Sciences
Universiti Utara Malaysia
Sintok, Kedah, Malaysia

ABSTRACT

In this paper, the use of TF-IDF stands for (term frequency-inverse document frequency) is discussed in examining the relevance of key-words to documents in corpus. The study is focused on how the algorithm can be applied on number of documents. First, the working principle and steps which should be followed for implementation of TF-IDF are elaborated. Secondly, in order to verify the findings from executing the algorithm, results are presented, then strengths and weaknesses of TD-IDF algorithm are compared. This paper also talked about how such weaknesses can be tackled. Finally, the work is summarized and the future research directions are discussed.

Text Mining

Text Analytics

Keywords

TF-IDF, Data Mining, Relevance of Words to Documents

1. INTRODUCTION

The processing of structured or semi—structured data in all organizations is becoming very difficult as the data has been increased tremendously [1], [2]. There are many techniques or algorithms that can be used to process data but this study is focused on one of those, known as TF-IDF. TF-IDF is a numerical statistic that shows the relevance of keywords to some specific documents or it can be said that, it provides those keywords, using which some specific documents can be identified or categorized. For example, a blogger is running a blog with hundreds of contributors and he just hired an internee whose main task is to add new blog posts on daily basis. It has been observed that most of the times internees does not take care of tags due to which many blog posts are not categorized. This is one of the ideal condition for applying TF-IDF algorithm, which can identify the tags automatically for the bloggers. It will save plenty of time for bloggers and internees, as they would not need to take care of tags [3].

The paper organization is as follows: In section 2, the background of TF-IDF algorithm will be discussed, then in section 3, the procedure and research method will be discussed. In section 4, implementation details will be explained for TF-IDF along its results, then section 5 will discuss the limitations of the algorithm and its related work and section 6, will elaborate how those limitations can be resolved through some new techniques. Finally, Section 7 will conclude the work and will discuss the future prospects.

2. BACKGROUND

2.1 Term Frequency (TF)

TF-IDF is a combination of two different words i.e. Term Frequency and Inverse Document Frequency. First, the term “term frequency” will be discussed. TF is used to measure that how many times a term is present in a document [4]. Let’s suppose, we have a document “T1” containing 5000 words and the word “Alpha” is present in the document exactly 10 times. It is very well known fact that, the total length of documents can vary from very small to large, so it is a possibility that any term may occur more frequently in large documents in comparison to small documents. So, to rectify this issue, the occurrence of any term in a document is divided by the total terms present in that document, to find the term frequency. So, in this case the term frequency of the word “Alpha” in the document “T1” will be

TF = 10/5000 = 0.002

2.2 Inverse Document Frequency (IDF)

Now, inverse document frequency will be discussed. When the term frequency of a document is calculated, it can be observed that the algorithm treats all keywords equally, doesn’t matter if it is a stop word like “of”, which is incorrect. All keywords have different importance. Let’s say, the stop word “of” is present in a document 2000 times but it is of no use or has a very less significance, that is exactly what IDF is for. The inverse document frequency assigns lower weight to frequent words and assigns greater weight for the words that are infrequent. For example, we have 10 documents and the term “technology” is present in 5 of those documents, so the inverse document frequency can be calculated as [4]

IDF = log_e (10/5) = 0.3010

2.3 Term Frequency - Inverse Document Frequency (TF-IDF)

From Section 2.1 and 2.2, it is understood that, the greater or higher occurrence of a word in documents will give higher term frequency and the less occurrence of word in documents will yield higher importance (IDF) for that keyword searched in particular document. TF-IDF is nothing, but just the multiplication of term frequency (TF) and inverse document frequency (IDF). We have already calculated TF and IDF in Section 2.1 and 2.2, respectively. To calculate the TF-IDF we can do as [4]

TF-IDF = 0.002*0.3010 = 0.000602

3. PROCEDURE & RESEARCH METHOD

3.1 Data Collection

The data was collected from 20 different random websites that belongs to 4 different domains. It was simple website’s content that is used in this study which is available to general public to consume for free over the internet.

Table 1. Domains & Websites

No.	**Domains**	**Website's Count**
1	.biz	5
2	.com	5
3	.edu	5
4	.org	5
Total	4	20

3.2 Data Preprocessing

Data is collected from websites hence collected data contained HTML/CSS which was not useful hence it was completely removed first. Secondly, the data also contained many stop words which is never meaningful or useful in this context as explained in Section 2.2. Hence in order to filter those stop words, a list of 500 stop words was used first which filtered the data and removed all stop words from it [1], [4], [5]. A large list of stop words can easily be obtained from many blogs and websites where it is available for free for general public to consume.

3.3 Design

In this study, there are a few steps that has to be followed in order to successfully implement the TF-IDF algorithm.

First of all, data has to be fetched from websites. Secondly, in preprocessing phase, one has to look for HTML/CSS and stop words and remove all of them as they are unnecessary has no importance in this scenario. Third, one need to count total number of words and their occurrences in all documents. Once these steps are performed, one can apply Term Frequency formula to calculate TF as discussed in Section 2.1 [1], [4], [6].

After calculating TF, one has to check, if each word is found in every document or not and has to count total number of documents in hand. Once these steps are completed, one can apply Inverse Document Frequency (IDF) formula to calculate IDF as discussed in Section 2.2 [4].

In the last, after obtaining TF and IDF, one can easily calculate TF-IDF by applying its formula as described in Section 2.3. The algorithm can be implemented in any programming language of your choice. For this paper, it was implemented using PHP for the sake of simplicity [4].

The TF-IDF process can be observed using a diagram here which is showing all major and minor steps that has to be taken in order to successfully implement the algorithm using computer programming.

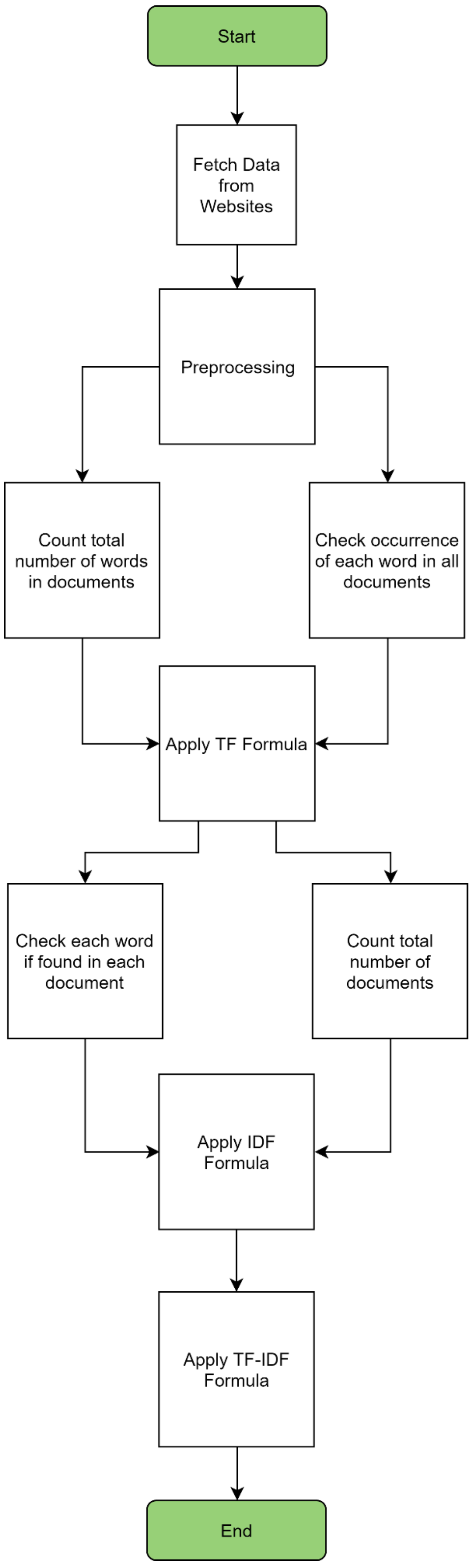

Start
Fetch Data from Websites
Preprocessing
Count total number of words in documents
Check occurrence of each word in all documents
Apply TF Formula
Check each word if found in each document
Count total number of documents
Apply IDF Formula
Apply TF-IDF Formula
End

Fig 1: TF-IDF Process

The Fig 1. Should be followed from top to bottom in order to implement TF-IDF. The process is quite simple but one really need to take special care on data preprocessing as it is very important in order to achieve accurate results.

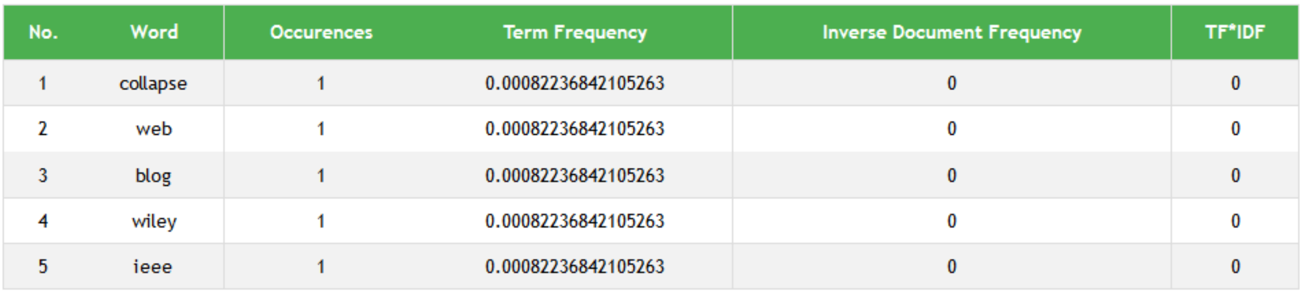
4. IMPLEMENTATION & DISCUSSION

The algorithm is implemented in PHP hence it can be used via a web browser. The interface is very easy to use where a user has to select document by clicking on browse button. After providing the document, the program will execute all the steps as mentioned in Fig 1 and will give output as shown below in Fig 2. The program gives serial number, word, their occurrences, term frequency, inverse document frequency and

Finally the TF-IDF on which this study is focused on.

the thing that should be noticed here is, one can sort the output of algorithm either in ascending order or descending order based on their occurrences or their TF-IDF score so that the keywords having greater occurrences or greater TF-IDF score would come on the top in decreasing order or the keywords having lower occurrences or lower TF-IDF score would come on the top in increasing order. That can really help in analyzing or slicing the data to generate reports or visualizations.

The program can be executed with minimum, a few microseconds time to a few seconds or a minute, depending on the size of the provided dataset.



No.	Word	Occurences	Term Frequency	Inverse Document Frequency	TF*IDF
1	collapse	1	0.00082236842105263	0	0
2	web	1	0.00082236842105263	0	0
3	blog	1	0.00082236842105263	0	0
4	wiley	1	0.00082236842105263	0	0
5	ieee	1	0.00082236842105263	0	0

Fig 2: TF-IDF Program interface after providing a dummy documents

Table 2. Results by running algorithm on three documents from each domain

Domain	**Keyword Rank (TF-IDF)**			
	Keywords	**TF**	**IDF**	**TF-IDF**
.biz	Parts	0.02189781	0.47712125	0.01044791
	Garden	0.02120141	0.47712125	0.01011564
.com	Presidential	0.00471698	0.47712125	0.00225057
	Held	0.002105263	0.47712125	0.00100446
.edu	Years	0.080519480	0.47712125	0.03841755
	Ms	0.038961038	0.47712125	0.01858913
.org	Marking	0.014084507	0.47712125	0.00672001
	Scholarships	0.017414965	0.17609125	0.00306662

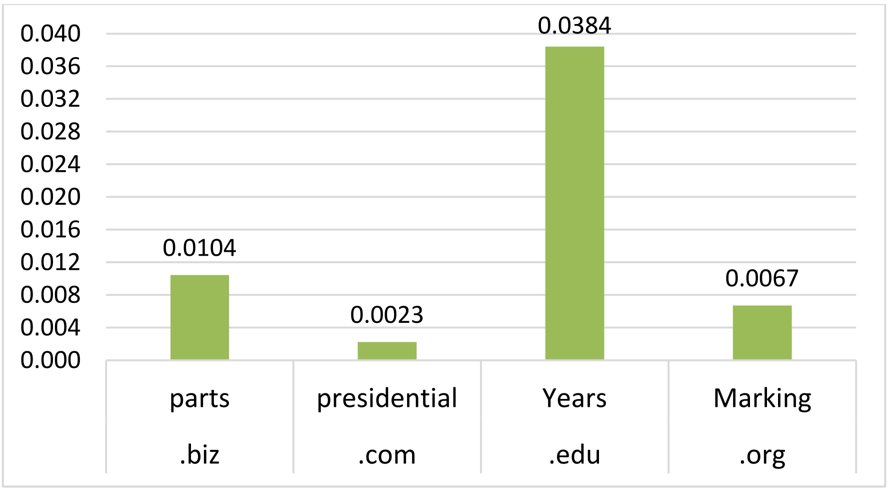


Fig 3: Top keywords as per their TF-IDF Score from 3 documents each for all domains

Table 2. Shows top two keywords according to highest TF-IDF score for three documents only from each domain. The data is fetched from the PHP program after running it on dataset collected from all domains as shown in Fig 2. Table 2. Depicts the fact that three documents of .biz domain that were selected, the most relevant keywords that can describe those documents are “parts” and “garden”. Similarly, in three documents of .com domain, the top two words are “presidential” and “held”, also in .edu domain the top keywords are “years” and “Ms”, which shows that these keywords can describe those documents and can be used as tags to categorize the content. The .org domain also has “Marking” and “Scholarships” keywords for the same purpose.

Fig 3. Shows only the top one keyword from each domain for three documents. It can be seen here that, the highest TF-IDF score is of the keyword “Years” that belongs to “.edu” domain. The second highest TF-IDF score is of the keyword “parts” that belongs to “.biz” domain and similarly the third position goes to “Marking” which belongs to “.org” domain.

The algorithm was tested again, this time with all documents that belongs to all domains

Table 3. Results Top 12 Keywords from all documents

Domains	**Keywords**	**TF-IDF**
.edu	Nts	0.07120161
.edu	Islamabad	0.06181321
.edu	Karachi	0.05132980
.edu	Bahria	0.04534312
.edu	Campus	0.04223501
.edu	Years	0.04122134
.com	Product	0.03543481
.biz	Goods	0.03212431
.edu	University	0.03012134
.com	Equipments	0.02112122
.org	Engineering	0.01233121
.org	Police	0.01121121

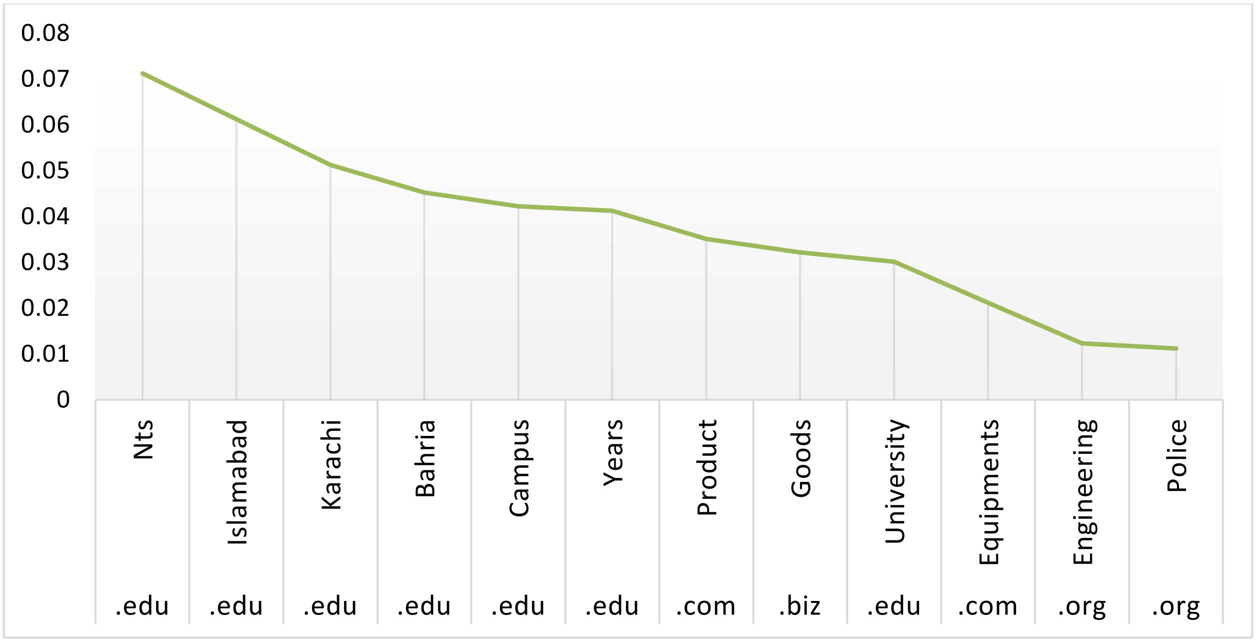


Fig 4: Top keywords as per their TF-IDF Score from all documents for all domains

Table 3. Shows top 12 keywords as per their highest TF-IDF score achieved from all documents and domains. In Fig 4, it can be observed that, when content was processed from all domains such as .biz, .com, .edu and .org, the most important and relevant keywords are displayed as a result in sorted form, means that, the keywords with highest TF-IDF score is on top and the lowest is at the end. It proves the fact that in TF-IDF algorithm you get results from most relevant to most

irrelevant keywords [3] which is very important, if one has to analyze the data or needs to generate tags for specifying the category of some document or blog post.

5. LIMITATIONS & RELATED WORK

There are some limitations of TF-IDF algorithm that needs to be addressed. The major constraint of TF-IDF is, the algorithm cannot identify the words even with a slight change in it's tense, for example, the algorithm will treat "go" and "goes" as two different independent words, similarly, it will treat "play" and "playing", "mark" and "marking", "year" and "years" as different words. Due to this limitation, when TF-IDF algorithm is applied, sometimes it gives some unexpected results [7]. Another limitation of TF-IDF is, it cannot check the semantic of the text in documents and due to this fact, it is only useful until lexical level. It is also unable to check the co-occurrences of words. There are many techniques that can be used to improve the performance and accuracy as discussed by [8], such as Decision Trees, Pattern or rule based classifiers, SVM classifiers, Neural Network classifiers and Bayesian classifiers. Another author [9] has also detected defect in standard TF-IDF that it is not effective if the text that needs to be classified is not uniform, so the author has proposed an improved TF-IDF algorithm to deal with that situation. Another author [10] has mixed TF-IDF with Naïve Bayes for proper classification while considering the relationships between classes.

6. SOLUTIONS

With the passage of time, new algorithms are coming up that resolves some limitations of older algorithms. For example, stemming process can be used to overcome the issues of TF-IDF not being able to identify that the "play" and "plays" are basically the same words [5]. The stemming process is basically used for conflating different forms of any particular word such as "play" and "plays" or "played" into a single and more generic representation such as "play". Secondly, the stop words can be added as much as possible so that the words that are not of any value such as "the" or "a" are filtered and removed before the data processing [5]. This will ensure to some extent, that you are getting useful words as output.

7. CONCLUSION

TF-IDF algorithm is easy to implement and is very powerful but one cannot neglect its limitations. In today's world of big data, world requires some new techniques for data processing, before analysis is performed. Many researchers has proposed an improved form of TF-IDF algorithm known as Adaptive TF-IDF. The proposed algorithm incorporated the hill-climbing for boosting the performance. A variant of TF-IDF has also been observed that can be applied in cross-language by using statistical translation. Genetic algorithms have also been put in work to improve the TF-IDF, as the natural genetic concepts of cross over, and mutation was applied programmatically, but it did not see the light of sun, as there was very slight improvement in performance. Search engine giants like Google has adapted latest algorithms such as PageRank for bringing out the most relevant results when a user place a query. In future research, world is going to witness some new techniques that can overcome the limitations of TF-IDF, so that the query retrieval can be more accurate. TF-IDF can be combined with other techniques such as Naïve Bayes to get even better results.

8. ACKNOWLEDGMENT

The authors wish to thank all the respected professors who helped during experiment, development and writing phase of this paper.

9. REFERENCES

[1] Bafna, P., Pramod, D., and Vaidya, A. (2016). "Document clustering: TF-IDF approach," International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), Chennai, 2016, pp. 61-66

[2] Trstenjak, B., Mikac, S., & Donko, D. (2014). "KNN with TF-IDF based framework for text categorization" In Procedia Engineering. Vol. 69, pp. 1356–1364. Elsevier Ltd

[3] Gautam, J., & Kumar, E.L. (2013). "An Integrated and Improved Approach to Terms Weighting in Text Classification," International Journal of Computer Science Issues, Vol 10, Issue 1, No 1, January 2013

[4] Hakim, A. A., Erwin, A., Eng, K. I., Galinium, M., & Muliady, W. (2015). "Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency (TF-IDF) approach," 6th International Conference on Information Technology and Electrical Engineering: Leveraging Research and Technology, (ICITEE), 2014

[5] Gurusamy, V., & Kannan, S. (2014). "Preprocessing Techniques for Text Mining," RTRICS, pp. 7-16

[6] Nam, S., and Kim, K. (2017). "Monitoring Newly Adopted Technologies Using Keyword Based Analysis of Cited Patents," IEEE Access, vol. 5, pp. 23086-23091

[7] Ramos, J. (2003). "Using TF-IDF to Determine Word Relevance in Document Queries," Proceedings of the First Instructional Conference on Machine Learning, pp. 1–4

[8] Santhanakumar, M., and Columbus, C.C. (2015). "Various Improved TFIDF Schemes for Term Weighing in text Categorization: A Survey," International Journal of Applied Engineering Research, vol. 10, no. 14, pp. 11905-11910

[9] Dai, W. (2018). "Improvement and Implementation of Feature Weighting Algorithm TF-IDF in Text Classification," International Conference on Network, Communication, Computer Engineering (NCCE 2018), vol. 147

[10] Fan, H., and Qin, Y. (2018). "Research on Text Classification Based on Improved TF-IDF Algorithm," International Conference on Network, Communication, Computer Engineering (NCCE 2018), vol. 147