International Journal of Advanced Computer Research
ISSN (Print): 2249-7277 ISSN (Online): 2277-7970
Volume-5 Issue-21 December-2015

Improved Vector Space Model TF/IDF Using Lexical Relations

Minh Chau Huynh, Pham Duy Thanh Le and Trong Hai Duong*
International University-Vietnam National University HCMC, Vietnam

Received: 12-August-2015; Revised: 15-October-2015; Accepted: 17-October-2015

Abstract

Current vector space model, for instance TF/IDF, has not yet taken into account the relations between terms; it only combines the term frequency in a document and the inverse document frequency in whole database to identify importance-score (weight) of a term respect with the document. Here we discover lexical relations among terms in the document to improve the vector space model TF/IDF. The weight generated from TF/IDF for each term, which is improved by lexical relations among related terms in the document. We evaluate the proposed method using documents selected from Wikipedia. The result shown that the proposed method is significant effective.

Keywords

Sector space model, TF/IDF, Semantics, Information retrieval, Natural language processing.

1. Introduction

IR (Information Retrieval) systems are designed for finding similarity documents. Vector Space Model borrowed from Information Retrieval [1], any text can be seen as a vector in V-dimensional space of a document, a query, a sentence, a word, an entire encyclopaedia [2]. This model is an algebraic model which represents text documents as vectors of identifier, such as terms [3]. The elements of this vector express the level of important of a word/term and the raw frequency of this word in a document. This model represents text as the point in n-dimensional Euclidean space, each dimension corresponding to one of the set of words.

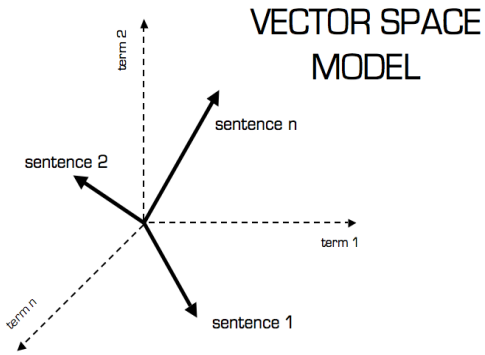


Figure 1: Vector space model

One of the most applications of Vector Space Model is TF/IDF. TF/IDF (Term Frequency/ Inverse Document Frequency) is one of basic methods to compute the weight of terms. There are many variants of TF/IDF. The following common variant was used, as found in [4]

$$weight_{t,d} = \begin{cases} \log(tf_{t,d}+1)\log\frac{n}{x_t} & \text{if } tf_{t,d} \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

TF (Term Frequency) which how calculate the raw frequency of a term occurs in a document. The different length of every document, which leads to a term, would appear much more times in long documents than shorter ones. Thus, the term frequency is often normalized by dividing the document length [5].

IDF (Inverse document frequency) is used to evaluate the level of importance of a term. While calculating TF, the importance of all terms is the same. However, many terms such as "a", "the", and "this", and so on, maybe occur frequently but have little importance. Thus we need to decline the weight of these terms while scale up the rare ones.

*Author for correspondence

With below TF/IDF formula, it means that the weight of term is highest when it occurs many times within a small number of documents.

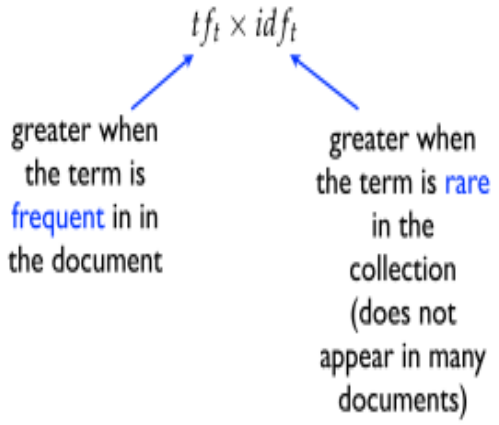


Figure 2: Illustrate meaning of TF/IDF [2]

However, TF/IDF maybe not the best method to evaluate the weight of terms in the document. The first problem, the relations among terms are not considered. For example, with the sentence: *laptop is one kinds of computer*. TF/IDF recognizes *laptop* and *computer* are different terms, but in the natural language, they have a relationship together. Another problem related to co-reference, considering a sentence: *Obama has had a speech in New York while he visited there three days*. In this case, *he* refers to *Obama*, which can improve the weight of *Obama* term. As in formal writing, the writer usually edits a document with more relations and co-references of terms together which avoid repeating the subjects. Therefore, in this paper, we base on relation and co-references of the natural language processing to propose an algorithm that enhances the weight of terms in the documents.

2. Principle Patterns

2.1 Co-reference resolution

2.1.1 Pronoun Co-reference

Pronoun co-reference determines the most likely antecedent for a pronoun in the text. The implemented algorithm for pronoun co-reference filters potential antecedents from the sentence containing the pronoun and from the preceding sentence. This filtering is based upon:

1. Grammatical constraints (e.g., a pronoun does not co-refer with a clausal co-argument; in the clause ". The PLA Commander promoted him", the direct object" him" cannot co-refer to the subject "The PLA Commander").
2. Mismatches of the features of number (e.g.," the commander" and "they"), gender ("the man" and "her"), and animacy ("the commander" and "it").
3. Mismatches in known semantic classes (e.g.," he "and "the U.N." where "the U.N." has been identified as an organization.

The potential antecedents that are not filtered according to these criteria are ranked according to salience. The numeric salience score is calculated based upon several features. These are:

1. Grammatical function (e.g., a clause subject is assigned a higher score than a clause direct object).
2. Semantic class (e.g., an identified person is assigned a higher score than a reference that has an unknown semantic class when evaluating potential antecedents for a singular personal pronoun).
3. Distance of the potential antecedent from the pronoun (the candidate references in the same sentence are scored higher than those in the preceding sentence).

2.1.2 Name Co-reference

Name co-reference determines when identified names in the text refer to the same entity. The name co-reference algorithm developed for the IAA-Cyc prototype checks for matches between one text reference and another. The matching criteria are based upon:

1. Aliases within the document that are indicated by parentheticals (e.g., "People's Liberation Army [PLA]").
2. Aliases stored in the database (e.g., "KFOR" as an alias for "Kosovo Force").
3. Normal forms derived according to syntactic features (e.g.,"U.N.Security Council" for" the U.N.Security Council").
4. Normal forms stored in the database (e.g., "Kosovo Force" for "KFOR").
5. Variants of names (e.g., "Videnov" for "Zhan Videnov").

6. Text matching, when the semantic types assigned the text references are consistent ("Arpad Goncz" matches "Arpad Goncz").

2.2 Lexico-Syntactic Patterns for Hypernymy

Three syntactic phenomena commonly encode the hypernymic proposition: verbs, appositive structures, and nominal modifications. Here, we consider appositive structures [7, 8] in which two noun phrases must be contiguous. Three kinds of appositive cues can then mark the second noun phrase: commas, parentheses, or lexical items (including, such as, particularly, and especially). For instance, consider a sentence (*). "*We identified the activities among students such as seminars, discusses, conferences"*, where *activity* is the hypernym of *seminar, discussion, conference*. The sentence is then transformed into the following lexico-syntactic expression:

(1a) NP_0 such as NP_1, NP_2. . . , (and ‖ or) NP_i i ≥ 1, where NP_i is phrase noun i, are such that they imply
(1b) for all NP_i, i ≥ 1, hypernym (head(NP_0), head(NP_i)) where head(NP_i) is head noun of NP_i.
In [7, 8] authors presented most of lexico-syntactic patterns for hyponymy. However, they have not analyzed NP yet to identify its head noun, so it often makes mistakes about finding hypernym relation between concepts.

3. Improved TF/IDF

3.1 Feature Vector of Document Using TF/IDF

Definition 2 (Feature Vector of Normal Document). Let $T^d = (t_1, t_2,\ldots, t_n)$ be the collect ion of all of key-words (or terms) of the document *d*. Term frequency *tf*(*d*, *t*) is defined as the number of occurrences of term *t* in document *d*. A set term frequency pairs, $P^d = \{(t,f)|t \in T^d; f >$ threshold$\}$, called the pattern of document. Given a pattern $P^d = \{(t_1, f_1); (t_2, f_2),\ldots, (t_m, f_m)\}$, let $\vec{d}$ be the feature vector of document d and let td be the collection of corresponding terms to the pattern, we have:

$$\vec{d} = (w_1, w_2,\ldots,w_m) \quad (1)$$

$$td = (t_1, t_2, \ldots, t_m) \quad (2)$$

where

$$w_i = \frac{f_i}{\sum_{j=1}^{m} f_j} * \log \frac{|D|}{|d:ti \in d|} \quad (3)$$

Definition 3 (Feature Vector of Hyperlink Document). Let $P^i = \{(t_1, f_1), (t_2, f_2), \ldots ,(t_m, f_m)\}$ be the pattern of the document *i* belonging to the set of hyperlink ds of the hyperlink document, *i*=1..*n*. A set term frequency pairs, $P^c = \sum_{n=1..n} P^i$, called the pattern of the ds. Let $\overrightarrow{ds}$ be the feature vector of the ds and let tds be the collection of corresponding terms to the pattern, we have:

$$\overrightarrow{ds} = (w_1; w_2, \ldots , w_k) \quad (4)$$

$$tds = (t_1, t_2,\ldots,t_k) \quad (5)$$

where

$$w_i = \frac{f_i}{\sum_{j=1}^{m} f_j} * \log \frac{|D|}{|d:ti \in d|} \quad (6)$$

3.2 Improving TF/IDF Using Pagerank Algorithm

According to our examination, an importance measurement of a term respecting to a document, which must take into account the contributions from all other related terms in the document. Here we propose an algorithm using pageranking to improve TF/IDF (for short, we call pagerank algorithm) by considering lexical relations among terms. It consists of two steps including *Initialization* and *Propagation*. In *Initialization*, first, a weighted graph $G = (V; E)$ is composed, where *V* is a set of nodes representing terms of the feature vector of a document generated by using TF/IDF. Each node associated with a weigh (see Session 2) and *E* is a set of edges (*V* * *V*) representing the relationship between the corresponding terms. The initial importance-score (weight) of each relationship is calculated by eq. 6.

In *Propagation*, the importance score of each term is improved by taking into account the contributions from all the other related terms in the document via characterization of four features of potentially important scores of the term and its relations, which drive the drifting stream of consciousness:

- A term is more important if there are more relations originating from the term.
- A term is more important if there is a relation originating from the term to a more important term.
- A term is more important if it has a higher relation weight to any other terms.
- A relation weight is higher if it originates from a more important term.

Let $r(c_i)$ be a function of an importance weight of term c_i, $r_i = r(c_i)$ be an importance weight value of the term c_i, $w(c_i, c_j)$ be a relation weight function, and $w_{ij} = w(c_i, c_j)$ be the weight of all relations from c_i to c_j. It is possible that there exists more than one relation from term c_i to term c_j. Therefore, r_j w_{ij} is the total importance value of all the relations from term c_i to term c_j. In fact, the basic idea underlying *Propagation* is similar to the idea of [6]; we present a similar recursive formula that computes the weight of a relation starting from term c_i to term c_j at the $(k+1)^{th}$ iteration (see eq. 7). The weight is proportional to the importance of c_i and is the inverse ratio of the sum of all the importance values of c_j's backward concepts at the $k + 1^{th}$ iteration.

$$W_{k+1}(c_i, c_j) = \frac{r_k(c_i)}{\sum_{t_J \epsilon B_J} r_k(t_J)} \quad (7)$$

And the recursive formulae are used to calculate the importance of term c_i at the $k +1^{th}$ iteration. The importance consists of two parts; one contributed by all the importance values of c_i's forward terms and the weight of relations from ci to the forward terms with probability $\propto$. The other is contributed by some independent jump probabilities (here$\frac{1}{v}$) with probability λ ;the formulation is then expressed as follows:

$$r_{k+1}(c_i) = \propto \frac{1}{v} + \lambda \sum_{c_j \epsilon F_i} w_{k+1}(c_i, c_j) r_k(c_j), \propto + \lambda = 1 \quad (8)$$

4. Experiment

4.1 Dataset

The dataset for experiment is collected from wikipedia.org. The dataset is classified into folders in which contains collected text-files following topics correspondly; for example Agriculture, Computer, Business, Fashion, Sport and so on (see Figure 3)

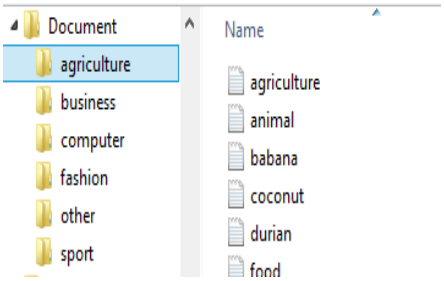


Figure 3: Data set

4.2 Outline Implementation

Input: A document, *d*

Output: weight of terms in document *d* using TF/IDF deal with coreference and pagerank algorithms.

Step 1: Calculate the weight of terms in document *d* using TF/IDF algorithm (for instance shown in *Figure* 4).

order	doc_name	term	tf	idf	tfidf
1489	computerprogramtxt	classified	0.00367647	2.52573	0.00928577
1523	computerprogramtxt	code	0.0147059	3.21888	0.0473364
1527	computerprogramtxt	collection	0.00367647	2.81341	0.0103434
1498	computerprogramtxt	commercial	0.00735294	2.30259	0.0169308
1540	computerprogramtxt	compiler	0.00367647	3.91202	0.0143824
1500	computerprogramtxt	computer	0.0404412	2.12026	0.085746
1507	computerprogramtxt	computerprogr...	0.00367647	3.91202	0.0143824
1481	computerprogramtxt	concerns	0.00367647	3.21888	0.0118341

Figure 4: The result of TF/IDF

Step 2: Find coreferences in document *d* with using Standford Core NPL [2]. The result is shown in *Figure* 5.

order	doc_name	coreference
2	computerprogramtxt	a computer ; 1 - the computer ; 3 -
3	computerprogramtxt	the program's instructions ; 2 - the instructions ; 3 -
4	computerprogramtxt	The same program in its human-readable source code form, from which executabl...
5	computerprogramtxt	computer programs ; 5 - computer programs ; 9 - Computer programs ; 10 -
6	computerprogramtxt	Computer source code ; 6 - Source code ; 7 - Source code ; 8 -
7	computerprogramtxt	an industrial or commercial product ; 12 - that ; 12 -
8	computerprogramtxt	it; e.g. Red Hat, Inc. or SUSE ; 13 -

Figure 5: The result of coreference

Step 3: Calculate the weight of terms after discovering the coreferences (see Figure 6).

order	doc_name	term	tf_coreference	idf	tfidf
1492	computerprogramtxt	classified	0.00367647	2.52573	0.00928577
1535	computerprogramtxt	code	0.0183824	3.21888	0.0591705
1539	computerprogramtxt	collection	0.00367647	2.81341	0.0103434
1510	computerprogramtxt	commercial	0.0110294	2.30259	0.0253962
1552	computerprogramtxt	compiler	0.00367647	3.91202	0.0143824
1512	computerprogramtxt	computer	0.0625	2.12026	0.132516
1519	computerprogramtxt	comput...	0.00367647	3.91202	0.0143824
1482	computerprogramtxt	concerns	0.00367647	3.21888	0.0118341

Figure 6: The result of TF/IDF after coreference

Step 4: Use rule of lexical relation and Standford Parser to find the terms have relationship together and store them in a graph database.

doc_name	subject	relation	object
computerprogramtxt	computer/NN;prog...	is/VBZ;	sequence/NN;instructions/NNS;task..
computerprogramtxt	Source/NN;code/N...	called/VBN;	program/NN;binary/NN;compiler/NN..
computerprogramtxt	Source/NN;code/N...	-LRB-/-LRB-;	program/NN;binary/NN;
computerprogramtxt	computer/NN;prog...	as/IN;	multitasking/NN;
computerprogramtxt	computer/NN;prog...	as/IN;	labor/NN;markets/NNS;profitability/..
computerprogramtxt	software/NN;softw...	as/IN;	product/NN;entity/NN;e.g./NNP;

Figure 7: The relations of terms

Step 5: Base on the graph of terms, we improve the weight of term by using pagerank algorithm.

4.3 Evaluation

Use rule of lexical relations and Stanford Parser to find the relationship of terms, we have the blow graph for computerproram.txt (see *Figure* 8).

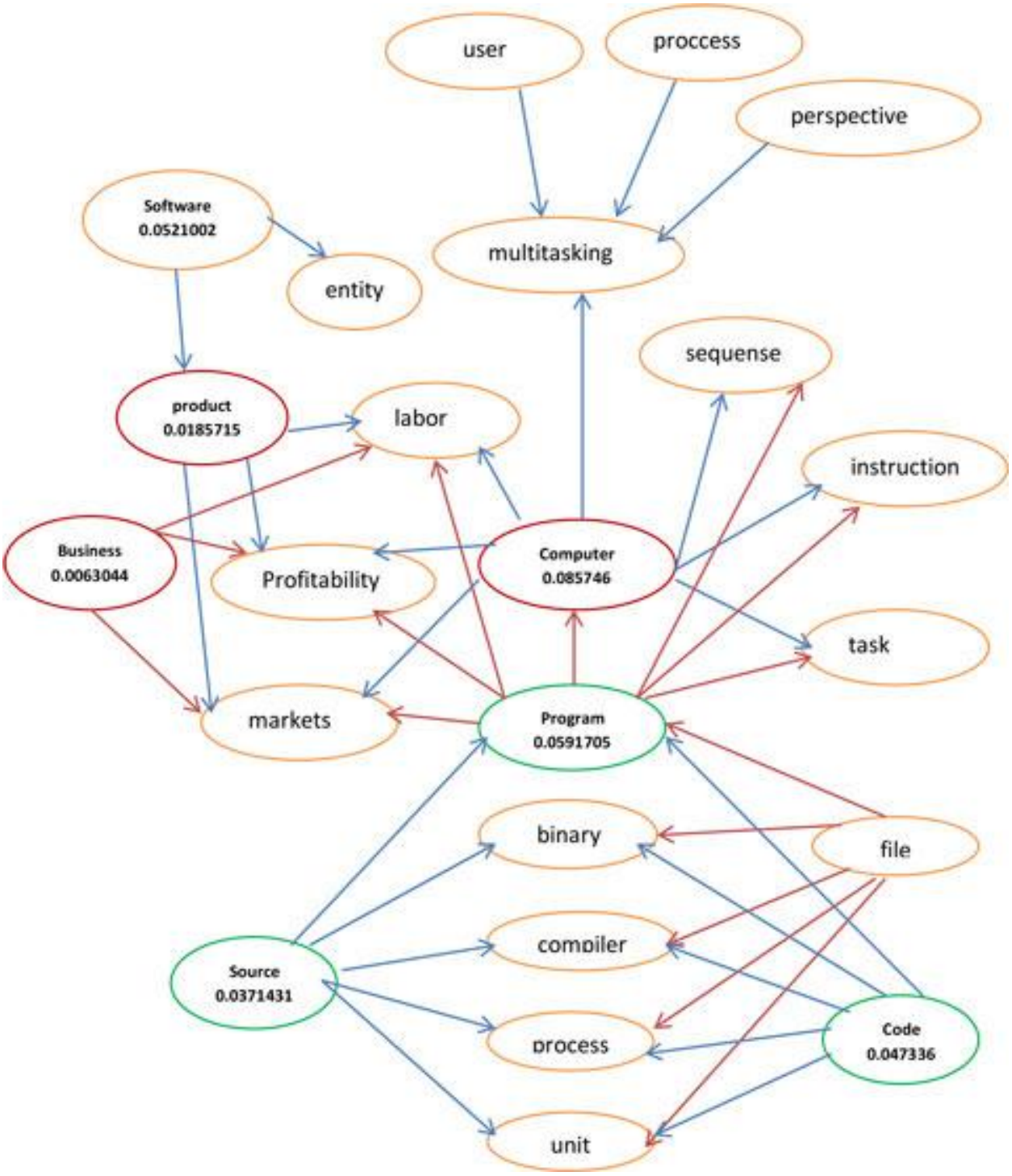


Figure 8: The relations between terms in computerproram.txt

Table 1: The weight of terms calculated by TF/IDF, coreference and pageranking algorithm

TERM	TF/IDF	TF/IDF-COREFERENCE	PAGERANK
computer	0.0857460	0.13251600	0.0622928
program	0.0591705	0.18934600	0.4362440
code	0.0473364	0.05917050	0.2289800
source	0.0371431	0.04642880	0.1796720
product	0.0185715	0.02785730	0.00000000 0000668
concerns	0.0118341	0.01183410	0.00000000 0000283
business	0.0063044	0.00630441	0.00000000 0000151
software	0.0620605	0.06206050	0.0620605
process	0.0092857	0.00928577	0.00000000 0005305

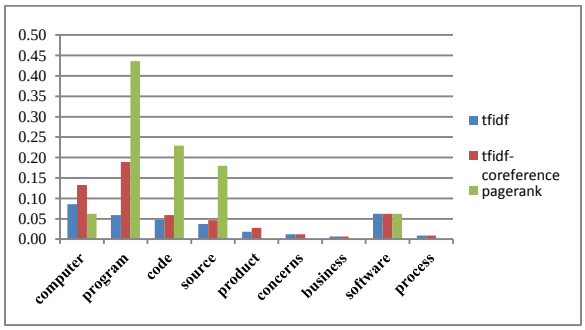
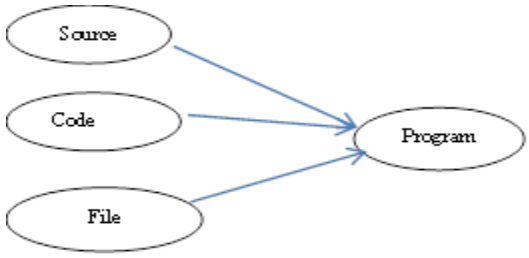


Figure 9: Comparison between the weight of terms calculated by TF/IDF, coreference and page ranking algorithms

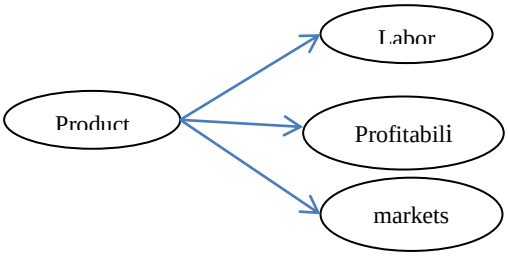
- With TF/IDF algorithm, the weight of "*Program*" term gains the second top approximately 0.05917. After use coreference algorithm its weight increases significantly. Especially, with pagerank algorithm, the weight of "*Program*" term is highest about 0.4362 (see *Figure* 9). There are two reasons making the weight of "*Program*" term go up:

- Firstly, there are more relations originating from "*Program*" term. Especially, it has many higher relation weight ("*Source*" term and "*Code*" term). A relation weight is higher if it originates from a more important term.



- Secondly, there is a relation originating from "*Program*" term to a more important term ("*Computer*" term).

- In contrast, with pagerank algorithm, the weight of "*Product*" term decreased significantly and approximately equal to 0. Because "*Product*" term has more relation originating from "*Product*" term to less important term.



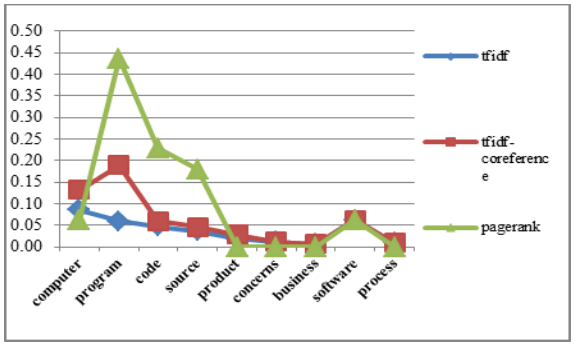


Figure 10: The difference between TF/IDF, Coreference and page ranking algorithms

Figure 10 shown that the pagerank method makes important terms to be clearer than TF/IDF method.

Coreference method can only improve the weight of terms, it cannot adjust the weight of terms for comfortable with the context. In contrast, pagerank algorithm not only improves the weight of important terms but also adjust the weight become more reasonable. For example, a document on Wikipedia.org which talks about "*Computer Program*". Deal with pagerank algorithm, the weight of terms such as: *Program*, *Source*, *Code* are increased. In contrast, the weight of "Product" term is decreased. This is reasonable with the context of document.

More experimental result is examined as follows:

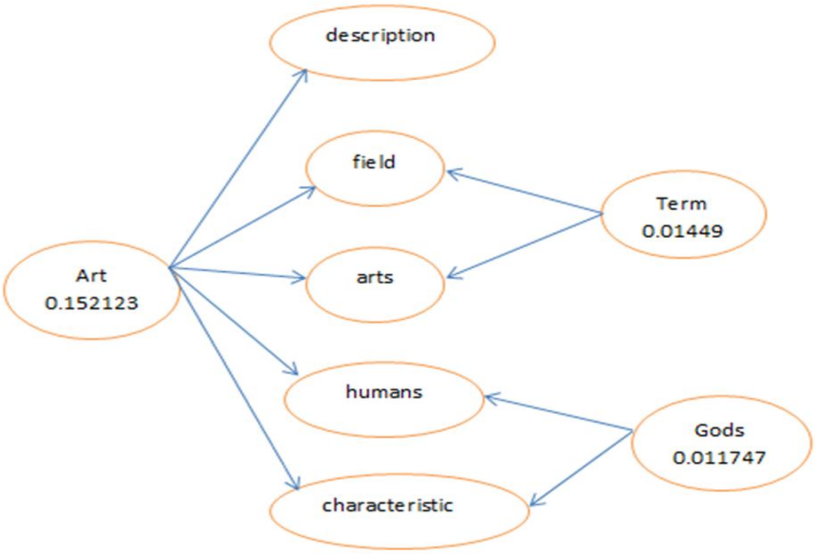


Figure 11: The graph show relations between terms in document art.txt

Table 2: The weight of terms calculated by TF/IDF, coreference and page ranking algorithms for document art.txt

TERM	TF/IDF	TF/IDF-COREFERENCE	PAGERANK
art	0.0624195	0.1521231	0.1956251
term	0.0048477	0.0144994	0.0032045
gods	0.0117832	0.0117478	0.0000181
description	0.0096954	0.0289989	0.0289989
field	0.0076076	0.0227543	0.0227543
arts	0.0969541	0.1159961	0.1159961
humans	0.0075847	0.0075847	0.0075847
characteristic	0.0234956	0.0234956	0.0234956

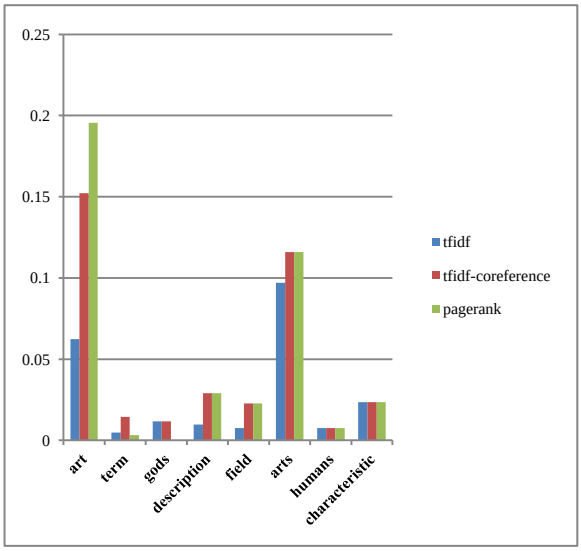


Figure 12: Comparison between the weight of terms use TF/IDF, coreference and page ranking algorithm of document art.txt

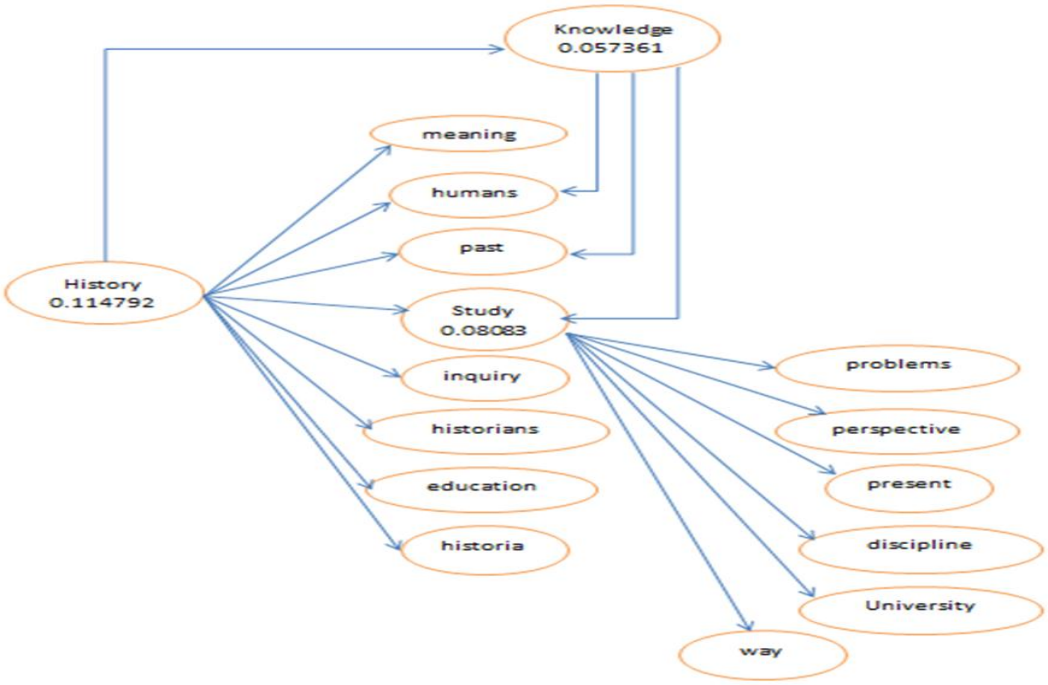


Figure 13: The relations of terms in document of history.txt

Table 3: The weight of terms calculated by TF/IDF, coreference and page ranking algorithm of document history.txt

Term	TF/IDF	TF/IDF-Coreference	Pagerank
History	0.0742769	0.114792	0.537597
Knowledge	0.0114722	0.057361	0.000000055838 5
Meaning	0.0057657 3	0.0288286	0.000000028063 4
Humans	0.0074068 3	0.00740683	0.00740683
Past	0.0222205	0.0666615	0.0666615
Study	0.0435245	0.0808312	0.00000317628
Inquiry	0.0114722	0.0688332	0.000000067006 2
Historians	0.0229444	0.0458888	0.000000006217 03
Education	0.0114722	0.0114722	0.000000000058 97
Historia	0.0114722	0.126194	0.000000122844 0
Problems	0.0094395 2	0.00943952	0.00943952
perspecti ve	0.009439 52	0.00943952	0.00943952
present	0.0074068 3	0.044441	0.044441
discipline	0.0458888	0.0803055	0.000000010879 8
University	0.0094395 2	0.018879	0.018879
way	0.0067524 5	0.00675245	0.00675245

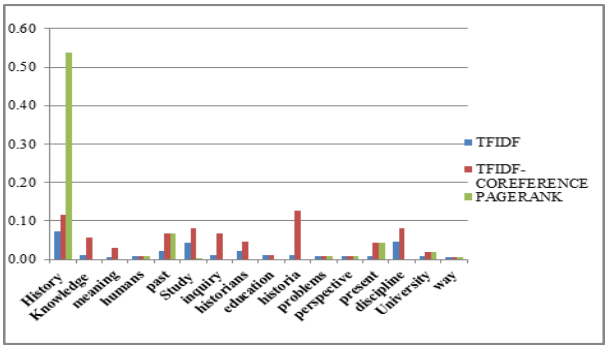


Figure 14: Comparison the weight of terms use TF/IDF, coreference and page ranking algorithm of document history.txt

Figure 15: The relations of terms in document of durian.txt

Table 4: The weight of terms calculate by TF/IDF, coreference and page ranking algorithm of document durian.txt

Term	TF/IDF	TF/IDF-Coreference	Pagerank
durian	0.0654184	0.143921	0.4498890
Tree	0.0130837	0.157004	0.1570040
Durio	0.0392511	0.170088	0.1700880
genus	0.00844725	0.00928577	0.0092858
fruit	0.0231029	0.107813	0.1495920
fragrance	0.0130837	0.0261674	0.0261674
Aroma	0.0130837	0.0130837	0.0130837
Species	0.0286756	0.0860267	0.0027703
regions	0.0094094	0.0094094	0.0094094
Market	0.0168945	0.0253418	0.0043995
centimetres	0.0261674	0.0261674	0.0261674
diameter	0.0107655	0.0107655	0.0107655
kilograms	0.0130837	0.0130837	0.0130837
shape	0.0107655	0.0215309	0.0215309
colour	0.0107655	0.0215309	0.0215309
husk	0.0322964	0.0538274	0.0538274
flesh	0.033789	0.0422363	0.0199786

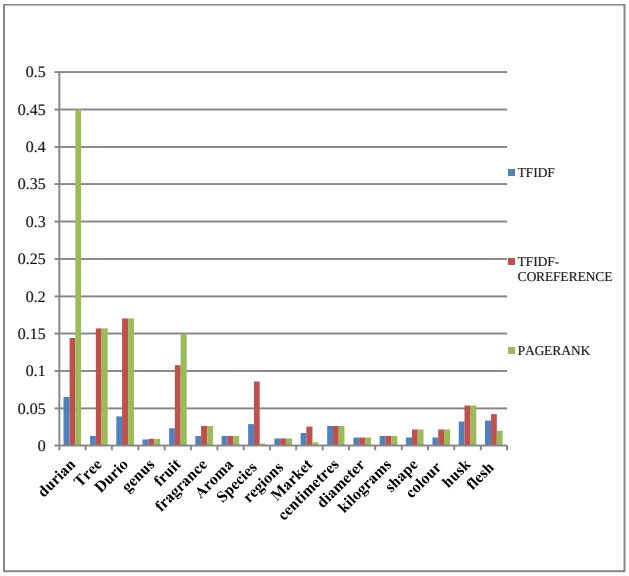


Figure 16: Comparison the weight of terms use TF/IDF, coreference and page ranking algorithm of document durian.txt

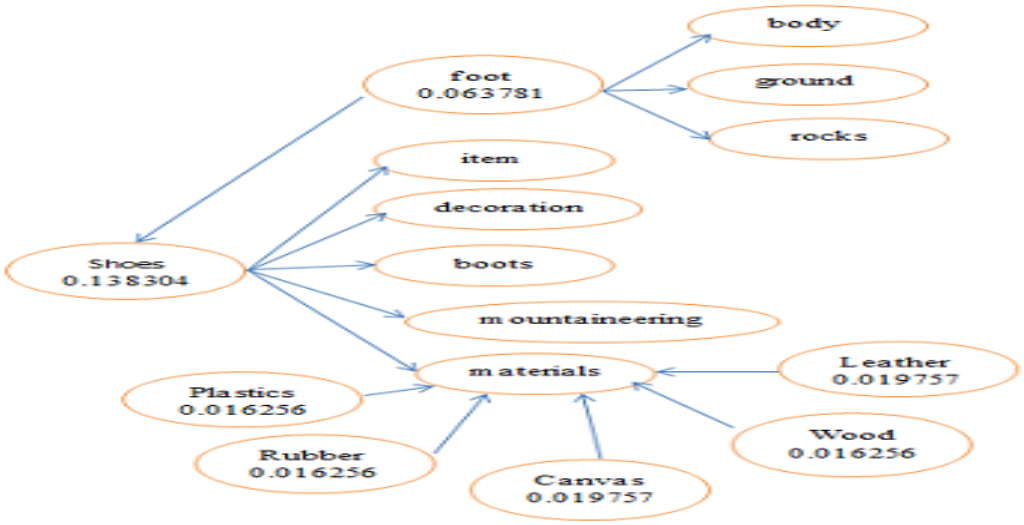


Figure 17: The graph show relations of terms in document of shoes.txt

Table 5: The weight of terms calculate by TF/IDF, coreference and page ranking algorithm of document shoes.txt

Term	TF/IDF	TF/IDF-Coreference	Pagerank
Shoes	0.138304	0.138304	0.707065
foot	0.0382686	0.063781	0.211705
body	0.0086606	0.0086606	0.0086606
Ground	0.0127562	0.0127562	0.0127562
Rocks	0.0197577	0.0197577	0.0197577
Item	0.0395154	0.0592731	0.0592731
Decoration	0.0197577	0.0197577	0.0197577
Boots	0.0197577	0.0197577	0.0197577
mountaineering	0.0197577	0.0197577	0.0197577
materials	0.0185109	0.0185109	0.0185109
Plastics	0.0162569	0.0162569	0.00000000000000164
Rubber	0.0162569	0.0162569	0.00000000000000164
Canvas	0.0197577	0.0197577	0.00000000000000200
Wood	0.0162569	0.0162569	0.00000000000000164
Leather	0.0197577	0.0197577	0.00000000000000200

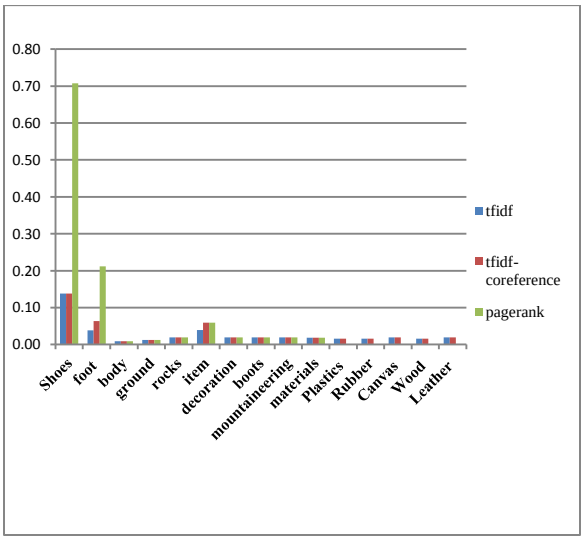


Figure 18: Comparison between the weight of terms use TF/IDF, coreference and page ranking algorithm of document shoes.txt

5. Related Work

A method using a neural network based language model that distinguishes and uses both local and global context via a joint training objective in order to enhance Vector-space models (VSM) [9]. Certainly, a method to improve vector space model using multilingual correlation was proposed. In particular, the independent VSM with different

languages are created and then projects them onto a common vector space. The latent semantic analysis serves as the monolingual VSM baseline [10]. Another approach, VSMs according to the type of matrix involved: term {document, word {context, and pair {pattern [11]. They believe that the choice of a particular matrix type is more fundamental than other choices, such as the particular linguistic processing or mathematical processing.

In the last few years, the Topic-based Vector Space Model (TVSM) [12] and the enhanced Topic-based Vector Space Model (eTVSM) [13] have been proposed. [14] keep on propose the first spam filtering model that uses an eTVSM to represent e-mail messages. They use an implementation of an eTSVM that applies the WordNet semantic ontology [15] for identifying synonym terms that share same interpretation.

6. Conclusion

We proposed a method to improve accuracy of TF/IDF by taking to account the lexical relations among terms in documents. To evaluate the method, tests have been conducted using dataset of Wikipedia with different kind of topics. *Figure* 10, 12, 14, 16, 18 shown that the TF/IDF with page ranking method clearly distinguishes weights of terms. TF/IDF with coreference method can only improve the weight of terms, it cannot adjust the weight of terms for comfortable with the context. In contrast, TF/IDF with page ranking not only improves the weight of important terms but also adjust the weight become more reasonable. The experimental result shown that the proposed method is a significant technique of computing importance-weight of terms belonging to documents. In future work, we will extract many more relations among terms to improve accuracy.

References

[1] Pascal Soucy, Guy W. Mineau, “Beyond TF/IDF Weighting for Text Categorization in the Vector Space Model”, IJCAI-05, pp 1136-1141.
[2] Jaime Arguello, “Vector Space Model” Information Retrieval September 25, 2013.
[3] Shuigeng Zhou, Songmao Zhang, George Karypis, “Advanced Data Mining and Applications”, 8th International Conference, ADMA 2012 Nanjing, China, December 2012 Proceedings, pp. 323.
[4] Yang, Yiming, and Xin Liu. "A re-examination of text categorization methods." Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1999.
[5] http://www.TF/IDF.com/
[6] W. Gang, L. Juanzi, F. Ling, W. Kehong, Identifying Potentially Important Concepts and Relations in an Ontology, In Proceedings of International Semantic Web Confer-ence’2008, Lecture Notes in Computer Science, 5318 (2008), 33-49.
[7] Hearst, M.A.: Automated Discovery of WordNet Relations, In: Fellbaum, C, (ed.) WordNet: An Electronic Lexical Database, pp. 131-151, MIT Press, Cambridge (1998).
[8] Nguyen, Ngoc Thanh. Advanced methods for inconsistent knowledge management. Springer Science & Business Media, 2007.
[9] Eric H. Huang, Richard Socher, Christopher D. Manning, Andrew Y. Ng,” Improving Word Representations via Global Context and Multiple Word Prototypes”, Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. 2012.
[10] Manaal Faruqui and Chris Dyer, “Improving Vector Space Word Representations Using Multilingual Correlation”, 2014.
[11] Turney, Peter D., and Patrick Pantel. "From frequency to meaning: Vector space models of semantics." Journal of artificial intelligence research 37.1 (2010): 141-188.
[12] Becker, J., & Kuropka, D. “Topic-based vector space model.”, In Proceedings of the 6th international conference on business information systems, pp. 7–12, 2003.
[13] Kuropka, Dominik. "Modelle zur Repräsentation natürlichsprachlicher Dokumente-Information-Filtering und-Retrieval mit relationalen Datenbanken." Advances in Information Systems and Management Science 10 (2004).
[14] Santos, Igor, et al. "Enhanced topic-based vector space model for semantics-aware spam filtering." Expert Systems with applications 39.1 (2012): 437-444.
[15] Miller, George, and Christiane Fellbaum. "Wordnet: An electronic lexical database." (1998).

Huynh Minh Chau. I got B.S degree at school of Computer and Information Engineering, Huflit University, 2008. His research interest focuses on big data and natural language processing. Currently, He is a master student at International University-Vietnam National University HCMC, Vietnam.

Pham Duy Thanh Le plays an important role as Information System Manager at Home Credit Vietnam; He got B.Sc., AUT University (2011) & M.Sc., International University (2014). With wide range of experience in terms of Data Warehouse, Business Intelligence, Six Sigma Methodology, ERP as well as Contact Center, it is crucial that his research passion virtually focuses on data mining, big data, and enterprise system.

Trong Hai Duong got M.Sc. and Ph.D. degree at school of Computer and Information Engineering, Inha University, 2012. His research interest focuses on ontology and semantic web, big data, smart data, and ecommerce systems. He has more than 40 publications. Currently, he works for International University- Vietnam National University HCMC.
Email: haiduongtrong@gmail.com