A New Term Ranking Method Based on Relation Extraction and Graph Model for Text Classification

Dat Huynh **Dat Tran** **Wanli Ma** **Dharmendra Sharma**

Faculty of Information Sciences and Engineering
University of Canberra
ACT 2601, Australia,
Email: {dat.huynh, dat.tran, wanli.ma, dharmendra.sharma}@canberra.edu.au

Abstract

Term frequency and document frequency are currently used to measure term significance in text classification. However, these measures cannot provide sufficient information to differentiate important terms. Thus, in this research, a new term ranking (weighting) approach for text classification will be proposed. The approach firstly is based on relations among terms to estimates the important levels of terms in a document. Secondly, the proposed approach provides a considerable representation for the text documents. The results from experiment show that with the same data in Wikipedia corpus the term weighting approach provides higher accuracy in comparison to the popular approaches based on term frequency.

Keywords: Text Representation, Term Weighting Approach, Relation Extraction, Graph Building Model, Graph Weighting Model, Text Classification.

1 Introduction

The task of text classification is to automatically assign single or multiple category labels to a new text document based on category models created after learning a set of training documents with correct category labels. Current text classification methods convert a text document into a relational tuple using the popular vector-space model to obtain a list of terms with corresponding frequencies. A term-by-frequency matrix, interpreted as a relational table, will be obtained to represent a collection of documents (Wang et al. 2005).

Term frequency *(tf)* has been used to measure term significance in a specific context (Robertson & Jones 1997) and to estimate the probabilistic distribution of features using maximum likelihood estimates. The more a term is encountered in a certain context, the more it contributes to the meaning of the context. Other approaches based on the combination between *tf* and inverse document frequency *(idf)* (Yang & Pedersen 1997, Joachims 1998, Yang & Liu 1999, Yu & Zhang 2009) have also been proposed to solve the problems of classifying text documents. However, with some abstract and complex corpus where the contents of text documents are much more equivalent in term of statistic information, those approaches cannot differentiate documents and achieve high classification results (Joachims 1998, Yang & Pedersen 1997).

To overcome this shortcoming, some approaches have been recently proposed to discover more relationships among terms that represent for a given document. Most of those approaches used graph model to connect the relationships and applied centrality algorithms to identify significant terms that represent the document context. Relationships could be extracted from particular collections such as WordNet or Wikipedia, in which connections between words are based on characteristics of word-senses or Wikipedia links (Wang et al. 2007, Gabrilovich & Markovitch 2007, Strube & Ponzetto 2006, Yeh et al. 2009, Hu et al. 2008).

Term co-occurrence (*tco*) is the most popular method to model relationships among terms. The relations are considered as parts of a graph, and a random walk algorithm is applied to rank important terms based on characteristics of the connections (Hassan & Banea 2006, Wang et al. 2007, 2005). Our investigation indicates that *tco*-based methods can provide more relationships among terms. In a recent work (Hassan & Banea 2006), any combinations of single nouns within a certain window are accepted as relationships. However, the contribution of those relations to the document in term of semantic aspects is still an open question. In order words, those combinations are more referable to the statistical relationships rather than the relationship in document contexts. The relations not only contribute statistical information to document representation, but also more importantly they contribute noises to the contexts of document representation in terms of semantic aspects.

Recognising the deficiency of those approaches, in this research, an alternative term weighting method is proposed, which not only discovers more meaningful relationships among terms from a short context, but also weights the importance of terms based on their participations on the global contexts.

The remaining of this paper is organised as follows. Section 2 presents the framework of term weighting approach. Section 3 explains in details the methodology for extracting relations between terms from a given text document. Section 4 introduces the procedures of taking advantages of relations to return list of term representatives. Section 5 shows how to apply the document representations to text categorisation tasks. Section 6 describes our experimental results. The conclusion and future work will be discussed in section 7.

2 The Proposed Term Weighting Framework

The proposed term weighting framework includes the following main phases, (*figure 1*):

1. *Relation Extraction:* Each document in a text corpus is processed to extract a set of relations representing the contents of that document.
2. *Graph Ranking:* A weighted and directed graph is constructed by connecting all extracted relations from the previous phase. A random walk algorithm is applied to estimate the levels of importance of terms as their weightings. A weighted list of terms is returned and is regarded as the representation of the given document.

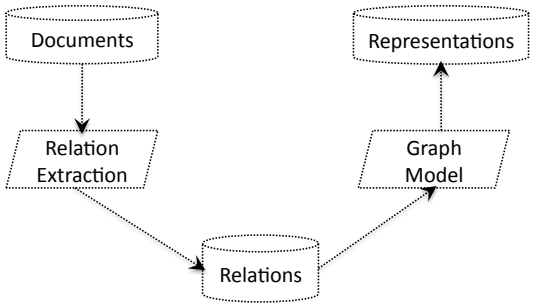

Figure 1: The proposed framework of extracting document representations from a text corpus

3 Relation Extraction

Relation extraction is an important research in text mining, which aims at extracting relationships between entities from text documents. The unsupervised relation extraction method (Banko et al. 2007) is used to extract relationships from web documents. Other methods extract relations based on particular kinds of patterns or seeding examples (Agichtein et al. 2000, Etzioni et al. 2004, Brin 1998). However, those methods extract relations from unstructured documents, where their efficiency can be considered in a case of working with a large number of input documents. Thus, to commit the task of extracting relations among terms, it is necessary to consider an alternative approach to extract relations for text classification.

In this section, a method of extracting relations from an input document based on syntactic analysis is presented. The approach considers roles of words in a sentence to take into account their built-in and hidden relations. For instance, from the sentence *"Antibiotics kill bacteria and are helpful in treating infections caused by these organisms"*, a list of relations will be obtained such as *(Antibiotic, kill, bacteria)*, *(Antibiotic, treat, infection)*, *(Antibiotic, treat infection cause, organism)*, and *(infection, cause, organism)*.

A relation is considered as a tuple $t = (e_i, r_{ij}, e_j)$, where e_i and e_j are strings denoted as terms, and r_{ij} is a string denoted as the relationship between them. *Figure 2* gives an example of tuples.

```
(Garlic, kill , bacteria)
(Garlic , inhibit , bacteria)
(Garlic, kill, virus)
(Garlic , prevent , disease)
```

Figure 2: Examples of tuples

3.1 A Framework for Extracting Relations

The relation extraction stage consists of three main steps: pre-processing documents, extracting tuples and optimising relations as shown in *figure 3*.

- *Pre-processing documents:* documents from the input corpus are pre-processed to extract sentences.
- *Extracting tuples:* an extractor applies a linguistic parser to analyse the syntactic structure of an input sentence and outputs a graph of linkages. The extractor then walks along the graph and applies an heuristic algorithm to extract raw tuples.
- *Optimising relations:* an optimiser receives all the raw tuples as its input and converts every single word into its simple form. Some non-essential words such as stop-words from tuples will be eliminated. The remaining tuples are regarded as a set of relations.

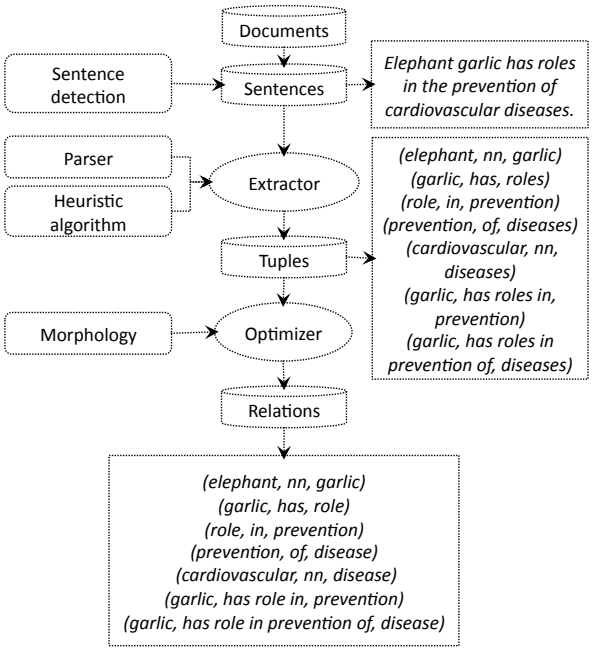

Figure 3: The proposed framework of extracting relations

3.2 An heuristic algorithm to extract relations

A linguistic parser[1] is used to analyse a sentence to produce a graph of linkages, in which relations among words are discovered. *Figure 4* shows the graph of linkages extracted from the following sentence *"Elephant garlic has a role in the prevention of cardiovascular disease"*.

[1]The Stanford parser http://nlp.stanford.edu/software/lex-parser.shtml

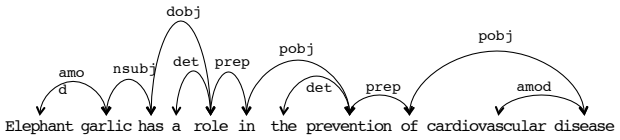

Figure 4: A linkage structure of a sentence

Moreover, the statistic information from Downey et al. (2005) has demonstrated that the majority sentences from English documents are categorised into certain kinds of sentence structures. Thus, the heuristic algorithm is designed to walk on the graph of linkages and extract raw tuples, based on those kinds of sentence structures.

The algorithm firstly scans a graph of the linkages from a sentence and identifies every single pairs of base noun[2] (e_i, e_j) with $i < j$. From each pair of noun phrases, if there is a shortest path (Bunescu & Mooney 2005) connecting from e_i to e_j, the algorithm will go along the path and identify a sequence of words between e_i and e_j. These words are considered as a potential relation r_{ij} to form the raw tuple $t = (e_i, r_{ij}, e_j)$. In the case that e_i and e_j are located next to each other, they should be connected by a direct connection. So, the relation r_{ij} is regarded as the name of the linkage (kinds of syntactic connection).

In short, if the raw tuples are passed through the following list of constrains, they will be retained to the next processing step. Here are some constraints to test the raw tuples

- e_i has to be a base noun
- e_j has to be a base noun
- r_{ij} has to be in a shortest path connecting e_i to e_j
- r_{ij} has to contain a verb or a preposition.
- w_k belonged to r_{ij} needs to match $i < k < j$.

After extracting the set of raw tuples from each document, components in each tuple is then optimised. First of all, all non-essential words will be eliminated such as adverbs, relative clause marker (who, whom, which, that, etc.), and stop-words from e_i and e_j. Then the morphology technique is used to convert all words in their simple forms such as the plural nouns into its single form, any kind of verb forms into its *"root"* word (Minnen et al. 2001). For instance, the noun phrase *"developing countries"* is converted to *"develop country"*, the verb phrase *have been working"* is converted to *"have be work"*. Once all wired tuples are eliminated, the remaining tuples are considered as a set of relations representing the document.

4 Graph Construction: Constructing, Weighting and Ranking Graph

Graph model is an alternative way to model information, which shows clearly the relationships among its vertices. It also groups the related information in a certain way, in which a centrality algorithm can take their best advantages. Recent approaches have used the model to select representations of a given text document. Firstly, the co-occurrence between words from the document is regarded as the relationships on the graph (Rada & Paul 2004). The graph model considers words as its vertices, and the term co-occurrence is measured as the weight of the relation. Secondly, taking the advantages of the universal corpus, Wikipedia, some authors have constructed the graph based on Wikipedia links. The graph connects all the concepts among Wikipedia documents. Under the light of the success of graph model to extract the representations of documents, we also propose a method to extract the document representation by taking the advantages of the extracted relations as well as the graph model.

4.1 Conducting Graph

A graph model is built to connect all extracted relations. Given a relation $t = (e_i, r_{ij}, e_j)$, where e_i and e_j are considered as vertices in the graph and r_{ij} is considered as the edge connecting between e_i and e_j. The weight of the edge w_{ij} is calculated based on the importance of r_{ij} in its documents or the relatedness between e_i and e_j in the corpus. In this paper, we are using the former one to measure the weightings of relations.

4.2 Weighting Graph

The weight w_{ij} is calculated based on two factors. Firstly, it is based on the frequency of a relation t in the document d. The higher redundancy of relation t is, the more important it is in the document d. Secondly, w_{ij} is based on the redundancy of relation t in the corpus. The redundancy of a tuple determines how valuable of that information from its document (Downey et al. 2005). So, w_{ij} is calculated as follows. Let $t = (e_i, r_{ij}, e_j)$ be a relation of d, and $e = (e_i, w_{ij}, e_j)$ be an edge of the graph. So w_{ij} is calculated as

$$w(r_{ij}) = freq(t, C) * rf(t, d) \quad (1)$$

$$rf(t, d) = \frac{freq(t, d)}{\sum_{i=1}^{|t:t \in d|} freq(t_i, d)} \quad (2)$$

where $freq(t, C)$ is the frequency of tuple t in the corpus C, $freq(t, d)$ is the frequency of tuple t in the document d, and $rf(t, d)$ is the normalised relation frequency value of the relation t in the document d.

A document from the corpus is represented as a directed weighted multi-graph, in which every single term is considered as a vertex of the graph. In order to weight the important levels of terms, we are using a centrality algorithm PageRank (Lawrence et al. 1998). The first reason of using PageRank for weighting terms is based on its original intuition. A page will have a high rank if there are many pages in the web pointing to it, or if there are some pages with high ranks pointing to it. Adapting the ideas for weighting important terms, a term is considered as importance if it participates the majority relations with other terms of the document, or if it has the relations to other important terms. Secondly, from a larger number of centrality weighting algorithms, PageRank is considered an outperformed method in evaluating importance information from the graph (Ravi & Rada 2007).

As a result, in order to use PageRank algorithm, every vertex from the graph needs to be treated as a webpage, the graph of relations needs to be

[2]Base noun is considered as a single noun or a main noun in a noun phrase. With topic/theme text classification, noun information is more informative than other kinds of words

converted into a directed and weighted multi-graph. Thus, every undirected edge $e = (e_i, w_{ij}, e_j)$ is converted to two directed edges $\overrightarrow{e} = (e_i, w_{ij}, e_j)$ and $\overleftarrow{e} = (e_i, w_{ij}, e_j)$. Then, the directed graph is passed through the PageRank as its input data and the output is returned as a set of vertices with their ranking scores. Thus, every vertex e_i has its ranking score pr_i, which is considered as the degree of significance of term e_i in the document d. As a result, a list of terms with their ranking values is the representative for the given document.

5 Applying Graph-based Ranking Approach to Text Classification

In the previous sections, we have discussed the novel method of extracting the representatives of a given document from a text corpus. Given a text document d_j from a corpus C, the list of n terms representatives of d_j is

$$d = \Big\{\big(w_1, pr(w_1, d)\big), \ldots, \big(w_n, pr(w_n, d)\big)\Big\} \quad (3)$$

where w_i is the text value of term i in the document d whereas $pr(w_i, d)$ is the weight of term w_i. A list of categories of the corpus C is

$$C = \{c_1, c_2, \ldots, c_m\} \quad (4)$$

5.1 Graph-based Ranking Approach and Inverse Document Frequency Measure (pr.idf)

The popular method based on term frequency is $tf.idf$, in which the weighting of a term w_i in a document d_j from the corpus C is calculated as:

$$tf.idf(w_i, d_j) = tf(w_i, d_j) * idf(w_i) \quad (5)$$

$$tf(w_i, d_j) = \frac{freq(w_i, d_j)}{\sum_{k=1}^{n} freq(w_k, d_j)} \quad (6)$$

$$idf(w_i) = \log\left(\frac{|C|}{|d : w_i \in d|}\right) \quad (7)$$

The idea of $tf.idf$ is that term frequency is represented for the importance of a term in a document, and the inverse document frequency is represented for the importance of the term in its corpus. Adapting with the idea of $tf.idf$, we propose $pr.idf$ term weighting measure, which takes the advantages of idf and pr. Instead of using tf, we have been calculating the weighting values based on graph model pr as the importance of the terms in the document.

Thus, the formula of weighting a term w_i of the document d_j in the corpus C as below:

$$pr.idf(w_i, d_j) = pr(w_i, d_j) * idf(w_i) \quad (8)$$

where $pr(w_i, d_j)$ is a importance value of w_i from the document d_j. The $pr.idf$ value of each term will be filled to the feature vector for classifying task.

5.2 Graph-based Ranking Approach and Term Category Dependence Measure (pr.tcd)

The idea of term category dependence measure (tcd) is that terms represented for a document are dependence to its categories. Recognising the benefits of term category dependency, a measure tcd has been proposed, which takes into account the degree of belonging of a term to a particular category. If a word occurs frequently in many documents of one class, and never or infrequency occurs in other classes, it is considered as a representative of the class if its ranking value from the document is also comparable. We suggest a measure of degree of belonging of the term w_i to the category c_i

$$tcd(w_i, c_j) = \frac{tf(w_i, c_j) * df(w_i, c_j)}{\sum_{k=1}^{m} (tf(w_i, c_k) * df(w_i, c_k))} \quad (9)$$

c_k is a categories of the corpus C

The combination of term ranking on documents (pr) and term category dependence (tcd) presents a new method for weighting a term w_i from a document d_j that belongs to a original category or predicted category c_k as follows:

$$pr.tcd(w_i, d_j) = pr(w_i, d_j) * tcd(w_i, c_k) \quad (10)$$

the $pr.tcd$ value of each term will be added to the feature vector for classification task.

6 Experiments

6.1 Comparison Term weighting methods

From the previous sections, we have presented our proposed term-weighting methods $pr.tcd$ and $pr.tcd$. It can be seen that $pr.idf$ is a combination of our calculation based on the graph pr and idf. The purpose of $pr.idf$ is to show how effective between tf and pr measures when making the comparison between $tf.idf$ and $pr.idf$. Moreover, when making the comparison between $pr.tcd$ and $pr.idf$, we can clearly see how effective among tcd and idf measures.

Similarly, the second combination $tf.tcd$ presented below gives another aspect between tf and pr when making the comparison between $tf.tcd$ and $pr.tcd$.

$$tf.tcd(w_i, d_j) = tf(w_i, d_j) * tcd(w_i, c_k) \quad (11)$$

With those methods based on the dependencies between terms and categories ($pr.tcd, tf.tcd$), the testing data does not have information of categories. Thus, a strategy is suggested to obtain the initial categories for calculating tcd values from the testing set. The initial categories are a predicted label set returned by using a text classifier to predict the labels of $tf.idf$-based feature vectors from the testing set. The classifier is enriched by $tf.idf$-based training model from the training set. Once the tcd values are identified for every single term of the testing set, the formulas (10) or (11) will be used to estimate the weighting of features.

6.2 Data set

The evaluation data set is the collection Wikipedia XML Corpus compiled by Ludovic & Patrick (2006). The reason to choose Wikipedia corpus as the evaluation data is that it is considered not only as universal unstructured text corpus containing very large numbers of multi-theme documents but also as a high standard grammar corpus.

There are many sub-collections of this corpus. We have chosen the English Single-Label Categorisation Collection, which provides each document that belongs to one single category. With the purpose of

demonstrating the effects of our approach, we randomly select part of the corpus for our experiment. As our approach is based on the context of sentences to extract the desirable information, we ignore documents containing uncompleted sentences. All documents are selected from the original Wikipedia corpus if their sizes are greater than $1kb$. As a result, after eliminating some categories that contain very small number of documents (less than 10 documents), we obtain a total of 8502 documents assigned to 53 categories. These documents are divided randomly and equally to form a training data set and a test data set including 4251 documents and 53 categories in each set. The number of documents for each larger category is 100, and the number of documents for each smaller category is 12.

The pre-processing procedures for the raw text from the training and test data are performed in the same way. From those methods based on term frequency ($tf.idf, tf.tcd$), the raw input text is firstly tokenised and eliminated stop-words. The remaining information is pushed to these term weighting methods to calculate the ranking of representation of documents. The outcome of these methods is the list of term representatives of documents. Before starting to evaluate the effectiveness of these methods by classification techniques, it is necessary to reduce the number of features. We are using the simplest method to eliminate the number of features: document frequency. We try not to use those terms as features if they appear in less than 3 different documents in the corpus[3]. By doing all the aforementioned eliminations, the number features of the term-based methods have been reduced from $103,408$ to $24,075$ for training set and $85,575$ from $16,191$ for test set, whereas the numbers of features from the graph-based methods were decreased from $73,529$ to $17,277$ for the training set and from $61,261$ to $11,346$ for the test set.

6.3 Classifiers

Support Vector Machines (Vapnik 1995) is a state-of-the-art machine learning approach based on decision plans. The algorithm defines the best hyper-plan, which separates set of points associated with different class labels with a maximum-margin. The unlabelled examples are then classified by deciding in which side of the hyper-surface they reside. The hyper-plan can be a simple linear plan or a non-linear plan such as polynomial, radial, or sigmoid. In our evaluation we used the linear kernel since it was proved to be as powerful as the other kernels when tested on text classification data sets (Yang & Liu 1999).

6.4 Performance & Evaluation measures

To evaluate the classification system we use the popular accuracy measure defined as the number of correct predictions divided with the number of evaluated examples. Four weighting models that need to be tested are $tf.idf$, $tf.tcd$, $pr.idf$ and $pr.tcd$.

Corpus	$tf.idf$	$tf.tcd$	$pr.idf$	$pr.tcd$
Wikipedia	72.7%	77.2%	73.8%	**81.8%**

Table 1: SVM results on Wikipedia corpus

By examining the SVM text classification results from *table 1* with four different term weighting models, it can be seen clearly that $pr.tcd$ model provides the highest accuracy and the $tf.idf$ model achives the lowest accuracy when testing in the same Wikipedia data.

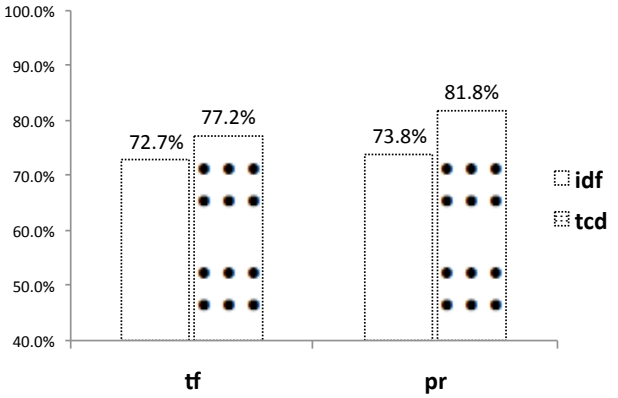

Figure 6: The chart shows the accuracy comparison of four different weight model *tf.idf, tf.tcd, pr.idf, pr.tcd*

6.4.1 Term Graph Weighting versus Term Frequency Weighting

The graph from *figure* 6 shows the comparison the effects of graph-based methods and term-based methods.

It is possible to conclude the graph model produces the more reliable text representation for the given text document than the traditional term frequency. Firstly, from the view of tcd, we are considering every pair methods $pr.tcd$ and $tf.tcd$ from the chart, the weighting method based on the graph model ($pr.tcd$) achieves 81.8%, which outperforms the one based on term frequency ($tf.tcd$) archiving 77.2%. Secondly, from the view of idf, when making the comparison between $tf.idf$ and $pr.idf$, it can be seen clearly that the graph-based method ($pr.idf$) dominates at 73.8%, which is higher at least 1% than the $tf.idf$ method (72.7%).

6.4.2 Inverse Document Frequency versus Term Category Dependency

The information from the chart of *figure 6* shows another view of information. It presents the comparison between the contribution of idf-based method and tcd-based methods to the performance of TC tasks.

The accuracy results have confirmed that all models taking the consideration of the dependency among terms and categories ($tf.tcd, pr.tcd$) yield the higher accuracy results than others based on document frequency ($tf.idf, pr.idf$) 77.2% vs. 72.2% and 81.8% vs 73.8%, respectively. It is also possible to conclude the tcd-based methods are more effective than the idf-based methods in text classification.

Moreover, the graph from *figure* 5 has also reflected the correlations among the proposed methods and the bottom-line ($tf.idf$) method for text classification tasks. There are 53 classes from the training and testing data for all tests. Every single line from the graph represents the accuracy of a particular weighing method in comparison with the others.

In short, the overall classification results have persuaded that the proposed methods provide outstanding results in comparison to the popular method ($tf.idf$) and can be considered as potential methods for further investigations.

[3]based on experiment reported by (Joachims 1998)

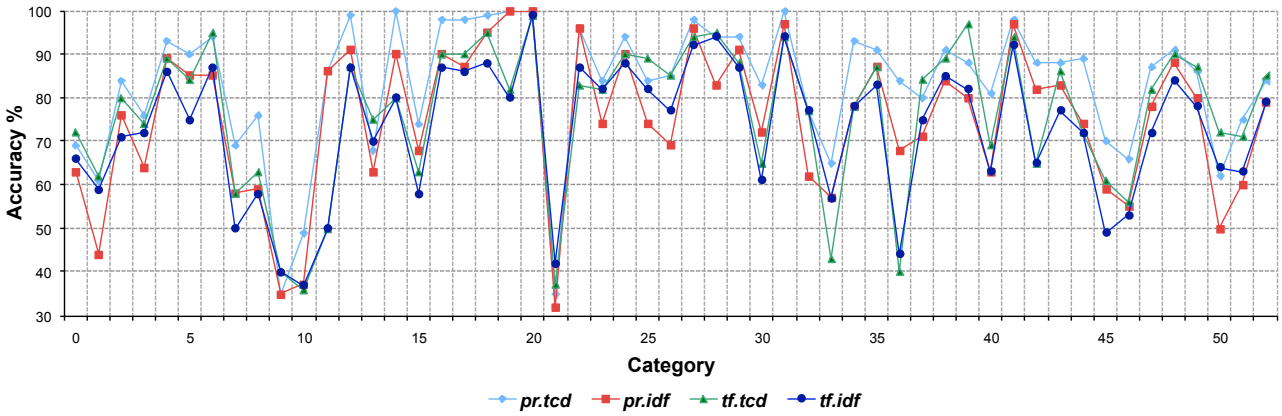

Figure 5: The chart shows correlation descriptions of four different term weighting methods *tf.idf, tf.tcd, pr.idf, pr.tcd* from the view of categories

7 Conclusion and future work

The paper has presented a method for document representation based on relation extraction and graph model, which improve accuracy of text classification in comparison to the popular term weighting methods. Our approach overcomes the lack of frequency information by self-creating the frequency based on the structure of text content. This is also the motivation for our further investigation on the benefits of relations on text classification as well as text mining.

We notice that although our approach uses the concept of relations, we still do not take the closed consideration on its semantic aspect, we only use it as a first attempt for getting more statistical information. For further investigation, we are more focusing on taking the semantic information from tuples and its connection from the graph to form representations of given documents. The expectation approaches can be used as objectives of semantic classifications.

References

Agichtein, E., Eskin, E. & Gravano, L. (2000), Combining strategies for extracting relations from text collections, *in* 'Proc. of the ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery'.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M. & Etzioni, O. (2007), Open information extraction from the web, *in* 'Proc. of IJCAI', pp. 2670–2676.

Brin, S. (1998), Extracting patterns and relations from the world wide web, *in* 'Proc. of the 1998 International Workshop on the Web and Databases', pp. 172–183.

Bunescu, R. C. & Mooney, R. J. (2005), A shortest path dependency kernel for relation extraction, *in* 'Proc. of Human Language Technology Conference and Conference on Empirical Methods in Natural Language', pp. 724–731.

Downey, D., Etzioni, O. & Soderland, S. (2005), A probabilistic model of redundancy in information extraction, *in* 'Proc. of the 19^{th} IJCAI', pp. 1034–1041.

Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D. & Yates, A. (2004), Web-scale information extraction in knowitall:(preliminary results), *in* 'Proceedings of the 13th international conference on World Wide Web', ACM, pp. 100–110.

Gabrilovich, E. & Markovitch, S. (2007), Computing semantic relatedness using wikipedia-based explicit semantic analysis, *in* 'Proc. of the 20^{th} IJCAI', pp. 1606–1611.

Hassan, S. & Banea, C. (2006), Random-walk term weighting for improved text classification, *in* 'Proc. of TextGraphs', pp. 53–60.

Hu, J., Fang, L., Cao, Y., Zeng, H.-J., Li, H., Yang, Q. & Chen, Z. (2008), Enhancing text clustering by leveraging wikipedia semantics, *in* 'Proc. of the 31^{st} Annual International ACM SIGIR Conference on Research and Development in Information Retrieval', pp. 179–186.

Joachims, T. (1998), Text categorisation with support vector machines: Learning with many relevant features, *in* 'Proc. of the 10^{th} ECML', pp. 137–142.

Lawrence, P., Sergey, B., Rajeev, M. & Terry, W. (1998), The pagerank citation ranking: Bringing order to the web, *in* 'Stanford Digital Library Technologies Project'.

Ludovic, D. & Patrick, G. (2006), The wikipedia xml corpus, *in* 'ACM SIGIR Forum', pp. 64–69.

Minnen, G., Carroll, J. & Pearce, D. (2001), Morphological processing of english, *in* 'Natural Language Engineering', pp. 207–223.

Rada, M. & Paul, T. (2004), Textrank: Bringing order into texts, *in* 'Proc. of the EMNLP'.

Ravi, S. & Rada, M. (2007), Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity, *in* 'Proc. of the ICSC', pp. 363–369.

Robertson, S. & Jones, K. S. (1997), Simple, proven approaches to text retrieval, Technical report, University of Cambridge.

Strube, M. & Ponzetto, S. P. (2006), Wikirelate! computing semantic relatedness using wikipedia, *in* 'Proc. of the 21^{st} AAAI', pp. 1419–1424.

Vapnik, V. N. (1995), The nature of statistical learning theory, *in* 'Springer'.

Wang, P., Hu, J., Zeng, H.-J., Chen, L. & Chen, Z. (2007), Improving text classification by using encyclopaedia knowledge, *in* 'The Seventh IEEE ICDM', pp. 332–341.

Wang, W., Do, D. B. & Lin, X. (2005), Term graph model for text classification, *in* 'Proc. of ADMA', pp. 19–30.

Yang, Y. & Liu, X. (1999), A re-examination of text categorisation methods, *in* 'Proc. of the 22^{nd} Annual International ACM SIGIR conference on Research and Development in Information Retrieval', pp. 42–49.

Yang, Y. & Pedersen, J. O. (1997), A comparative study on feature selection in text categorisation, *in* 'Proc. of the 14^{th} ICML', pp. 412–420.

Yeh, E., Ramage, D., Manning, C. D., Agirre, E. & Soroa, A. (2009), Wikiwalk: random walks on wikipedia for semantic relatedness, *in* 'Proc. of TextGraph-4', pp. 41–49.

Yu, S. & Zhang, J. (2009), A class core extraction method for text categorisation, *in* 'Proc. of the 6^{th} FSKD', pp. 3–7.