A Comparative Study using Vector Space Model with K-Nearest Neighbor on Text Categorization Data

Wa’el Musa Hadi
Department of Computer Information Systems
Arab Academy for Banking and Financial Sciences, Amman, Jordan
whadi81@students.aabfs.org

Fadi Thabtah
MIS Department Philadelphia University, Amman, Jordan
ffayez@philadelphia.edu.jo

Hussein Abdel-jaber
Department of Computing, University of Bradford, Bradford BD7 1DP, UK
habdelja@brad.ac.uk

Abstract— ***Text categorization is one of the well studied problems in data mining and information retrieval. Given a large quantity of documents in a data set where each document is associated with its corresponding category. Categorization involves building a model from classified documents, in order to classify previously unseen documents as accurately as possible. In this paper, we investigate variations of vector space model using inverse document frequency (IDF) and weighted inverse document frequency (WIDF). Experimental results against eight different data sets provide evidence that the Cosine Coefficient outperformed Jaccard and Dice Coefficient approaches with regards to F1 measure results, and the Cosine-based IDF achieved the highest average scores.***

Index Terms— **Data mining, Text categorization, Term weighting, Vector space model.**

I. Introduction

Text categorization (TC) is one of the important tasks in information retrieval (IR) and data mining. The problem of TC has been active for four decades [1], and recently attracted many researchers due to the large amount of documents available on the World Wide Web, in emails and in digital libraries. TC involves assigning text documents in a test data collection to one or more of the pre-defined classes/categories based on their content. Unlike manual classification, which consumes time and requires high accuracy, TC makes the classification process fast and more efficient since it automatically categorizes documents.

Many TC strategies from data mining and machine learning (ML) exist such as: decision trees [9], Support Vector Machine (SVM) [4], rule induction [8], and Neural Network [19]. In this paper we focus on a text similarity strategy, known as VSM in order to compute the similarity between incoming text (new test cases) and the pre-categorized text in the training data set. Generally, TC based on text similarity goes through two steps: Similarity measurement and classification assignment.

Term weighting is one of the known concepts in TC, which can be defined as a factor given to a term in order to reflect the importance of that term. There are many term weighting approaches, including, IDF and WIDF [16]. IDF and WIDF focus on terms occurrences inside a text corpus. WIDF distinguishes between two terms that have different occurrences, whereas, IDF treats both terms equally. In this paper, we compare different variations of VSM (Dice, Jaccard, Cosine) with KNN [21] algorithm using IDF and WIDF. The base of our comparison between the different implementations of KNN is the F1 measure [18]. In other words, we want to determine the best VSM, which if merged with KNN produces good results with reference to F1 measure results.

The organization of this paper is as follows, related works are discussed in Section 2. TC and similarity measures are described in Section 3. Section 4 is devoted to the experimental results and finally, conclusions and future works are given in Section 5.

II. Related Works

Since TC stands at the cross junction to modern IR and ML, Several research papers have focused on it but each of which has concentrated on one or more issues related to such task. There are some research works [2][3], which have focused on the different term weighting approaches related to TC such as Term Frequency (TF), WIDF, IDF, Chi-square [3] and ITF [7]. For example, the authors of [16] have achieved good improvement with reference to the retrieval accuracy using WIDF on Japanese language if compared to TF.IDF approach using KNN [21] and Bayesian Model [17]. Specifically, the KNN.WIDF implementation achieved 7.4% higher than that of the TF.IDF.

[22] have tested five categorization algorithms (SVM [4], KNN [21], NNet [19], LLSF [20] and NB based on Network [17]) on the Reuters-21578 TC data set. The results showed that SVM, KNN and LLSF outperformed NNet and NB-network when the number of positive training instances per category is less than ten. Further, all the methods performed well when the categories are well distributed in the training data

set.

The authors of [5] proposed a term weighting method called tf*rf, and compared their method using the traditional SVM, with other term weighting methods, i.e. (tf.x2, tf.ig, tf.or), on two widely used data sets from [23]. The experimental results showed that methods based on information theory, i.e. (tf.x^2, tf.ig, tf.or), perform poorly if compared with their proposed term-weighted method in terms of accuracy. Finally, a comprehensive comparative study [6] conducted on different term weighting methods using SVM showed that the term weighting method developed in [5] achieved better accuracy than other term weighting methods such as tf.ig and tf.or.

III. Text Categorization Problem

TC, also known as text classification, is the task of automatically sorting a set of documents into categories (or classes, or topics) from a predefined set. Such task is related to IR and ML communities. Automated text classification tools are attractive since they free organizations from the need of manual categorization of document, which can be too expensive, or simply not feasible given the constraints of the application or the number of documents involved [13].

TC involves many applications such as automated indexing of scientific articles according to predefined thesauri of technical terms, filing patents into patent directories, selective dissemination of information to information consumers, automated population of hierarchical catalogues of web resources, spam filtering, identification of document genre, authorship attribution, survey coding and even automated essay grading.

TC problem can be defined according to [12] as follows: let G denote the collection of categories which contain $\{g_1, g_2, \ldots, g_n\}$, let D denote the collection of documents and Q is an incoming text. Also, let R denote the set of classifiers for $D \times Q \rightarrow G$, each document d ε D is assigned a single class g that belongs to G. The goal is to find a classifier h ε H that maximizes the probability that r(d) = G for each test case (d, g). In TC, many term weighting methods can be used such as TF, IDF and WIDF, which we will discuss in the next subsection.

A. Term Weighting

Term weighting is one of the important issues in TC, which has been widely investigated in IR [10] [11]. Term weighting corresponds to a value given to a term in order to reflect the importance of that term in a document.

1) Term Frequency (TF)

One of the simplest term weighting methods that used to measure the importance of each term in a given document is TF [16]. Using this method, each term is assumed to have a value proportional to the number of times it occurs in a text. Generally, for a document d and a term t, the weight of t in d is given as:

$$W(d, t) = TF(d, t) \qquad (1)$$

TF can help in improving an IR and TC evaluation measure named recall [18] since frequent terms tend to appear in many documents, such terms have little discriminative power. Recall is the fraction of the relevant documents which has been retrieved and is represented in equation (11) according to Table II. To some extend, we can say that TF follows the normal distribution curve with regards to the importance of terms to the retrieval process, which means too much frequency or less frequency does not improve the retrieval process.

Fig. 1 demonstrates that the term frequencies in the interval [0, 5[and the interval] 15, 20] are too low, so we remove them from the term list. We also remove the stop words, which often has high frequencies. We keep the interval [5, 8.5] and the interval [11.5, 15] since the term frequencies are ideal for the retrieval process.

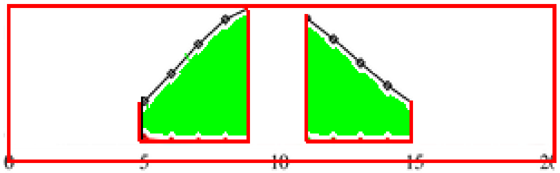

Fig. 1. Term frequency

2) Inverse Document Frequency (IDF)

TF reflects the importance of the term in a single document, however, what if we are interested in the frequency of a term in the set of documents. This is called the Inverse Document Frequency (IDF), meaning the importance of each term inversely proportional to the number of documents that contain that term [14]. Table I demonstrates that for a given corpus of documents, when a given term frequency increases within a document, the importance of that term decreases according to the IDF. In other words, when the term occurs in a small number of documents, this signifies it (when *n* equal ten). Whereas, when the term occurs frequently within a large number of documents, then it has insignificant importance according to IDF.

Table I. The relation between *N* documents and the importance of terms they contain.

N: Total no. of documents	***n***: no. of documents contain the term	IDF= Log(N/n)	Importance of the term
1000	10	2.000	Maximum
1000	20	1.699	

1000	40	1.399	
1000	80	1.097	
1000	160	0.795	
1000	320	0.494	
1000	640	0.190	Minimum

For a given N documents, if n documents contain the term t, IDF is given as follows:

$$\text{IDF (t)} = \log \text{(N/n)} \qquad (2)$$

Sometimes n is replaced by the document frequency (the number of documents that contain t), i.e. df(t). This approach follows Slaton's definition [11], which combined TF and IDF to weight the terms, and he showed that his approach gives better performance with reference to accuracy than IDF and TF. The product of TF and IDF is given in equation (3) below.

$$\text{W (d, t)} = \text{TF (t).IDF (t)} \qquad (3)$$

3) Weighted Inverse Document Frequency (WIDF)

One of the IDF drawbacks is that all documents containing a certain term are treated equally due to the binary counting. In other words, if a term "sea" occurred in 4 documents with different frequencies in each of these documents, the IDF does not consider the number of times in which "sea" has occurred in these 4 documents, rather it mainly considers the fact that "sea" has occurred. WIDF of a term t in document d is given by:

$$WIDF(d,t) = \frac{TF(d,t)}{\sum_{i \in D} TF(i,t)} \qquad (4)$$

Where TF (d, t) is the occurrence of t in d, and i ranges over the documents in the collection D. WIDF corresponds to the normalized term frequency over the collection. The weight of a term with reference to WIDF is given as:

$$\text{W (d, t)} = \text{WIDF (d, t)} \qquad (5)$$

B. Similarity Measurements

There are several well-known similarity techniques, such as: VSM, and Probabilistic Model (PM) [16]. In this paper we focus on VSM by adapting Cosine as shown in equation (6), Jaccard as shown in equation (7), and Dice as shown in equation (8).

$$Sim(Vi,Vj) = \frac{\sum_{k=1}^{m}(W_{ik} \times W_{jk})}{\sqrt{\sum_{k=1}^{m} W^2{}_{ik} \times \sum_{k=1}^{m} W^2{}_{jk}}} \qquad (6)$$

$$Sim(Vi,Vj) = \frac{\sum_{k=1}^{m}(W_{ik} \times W_{jk})}{\sum_{k=1}^{m} W^2{}_{ik} + \sum_{k=1}^{m} W^2{}_{jk} - \sum_{k=1}^{m}\left(W_{ik} \times W_{jk}\right)} \qquad (7)$$

$$Sim(Vi,Vj) = \frac{2\sum_{k=1}^{m}(W_{ik} \times W_{jk})}{\sum_{k=1}^{m} W^2{}_{ik} + \sum_{k=1}^{m} W^2{}_{jk}} \qquad (8)$$

Where Wik corresponds to the weight of the k-th element of the term vector V_i, i.e. pre-categorized documents, and W_{jk} is the weight of K-th element of the term vector V_j i.e. incoming text. The greater the value of $Sim(V_i,V_j)$, the more similar these two texts are.

C. KNN Algorithm

There are many approaches to assign category to incoming text such as [9][15][17]. In our paper, we implemented text-to-text comparison (TTC), which is also known as the k-nearest neighbor (KNN) [21]. KNN is a statistical classification approach, which has been intensively studied in pattern recognition over four decades. KNN has been successfully applied to TC problem, i.e. [22] [21], and showed promising results if compared with other statistical approaches such as Baysian based Network [17].

The KNN algorithm is quite simple: Given training and test documents, the algorithm finds the k-nearest neighbors among the training documents, and uses the categories of the k-neighbors to weight the category of the test document. The similarity scores of each neighbor document to the test document are used as a weight of the categories of the neighbor document. If several of the k-nearest-neighbors share a category, then the pre-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. By sorting the scores of the candidates' categories, a ranked list is obtained for the test document.

IV. Experiment Results

Experiments on the 20NewsGroups data sets (20NG) [23] using three TC techniques based on vector model similarity (Cosine, Jaccard, Dice) have been conducted. We used F1 evaluation measure as the base of our comparison, where F1 is computed based on the following equation:

$$F1 = \frac{2 * \Pr ecision * \operatorname{Re} call}{\operatorname{Re} call + \Pr ecision} \quad (9)$$

Precision and recall are widely used evaluation measures in IR and ML, where according to Table II,

$$\Pr ecision = \frac{X}{(X+Y)} \quad (10)$$

$$\operatorname{Re} call = \frac{X}{(X+Z)} \quad (11)$$

To explain precision and recall, let's say someone has 5 blue and 7 red tickets in a set and he submitted a query to retrieve the blue ones. If he retrieves 6 tickets where 4 of them are blue and 2 that are red, it means that he got 4 out of 5 blue (1 false negative) and 2 red (2 false positives). Based on these results, precision=4/6 (4 blue out of 6 retrieved tickets), and recall= 4/5 (4 blue out of 5 in the initial set).

The SVM methods considered in the experiments use similar strategy to classify incoming text i.e. K-nearest neighbors (KNN) [21]. We have several options to construct a text categorizer; we compared the above techniques using different term weighting methods, i.e. IDF, WIDF. All variations of the SVM-based KNN were implemented using VB.NET on 2.8 Pentium IV machine with 256 RAM. We have evaluated 8 selected data sets from the 20NG collection.

Table III represents the F1 results of the text categorizers generated against the 8 data sets, where in each data set we consider 100 documents arbitrary. We used 30 documents for each data set for testing purposes and the K parameter in the KNN algorithm was set to 5.

After analyzing Table III, we found out that there is consistency between Cosine based WIDF and Cosine based TF.IDF algorithms in which both of them outperformed Dice based TF.IDF, Dice based WIDF, Jaccard based TF.IDF, and Jaccard based WIDF. Particularly, Cosine based TF.IDF outperformed Dice based on TF.IDF, Dice based WIDF, Jaccard based TF.IDF, and Jaccard based WIDF on 6, 5, 6, and 5 data sets, respectively. The won-tied-loss records of Cosine based WIDF against Dice based TF.IDF, Dice based WIDF, Jaccard based TF.IDF, and Jaccard based WIDF are 6-1-1, 3-3-2, 6-1-1 and 3-3-2, respectively. There are similarities between 1) Dice based TF.IDF and Jaccard based TF.IDF and 2) Dice based WIDF and Jaccard based WIDF with respect to the average results of F1 measure.

TABLE II DOCUMENTS POSSIBLE SETS BASED ON A QUERY IN IR

Iteration	Relevant	Irrelevant
Documents Retrieved	X	Y
Documents not Retrieved	Z	W

Fig. 2 shows the F1 measure result for all variation of kNN algorithms. From Fig. 2, one can see that Cosine based TF.IDF achieved the highest F1 score on the Crypt data set. Moreover, Jaccard based TF.IDF and Dice TF.IDF achieved the lowest scores on ForSale data set.

TABLE III F1 RESULTS OF THE VSM IMPLEMENTATIONS WITH KNN

	Technique					
	Cosine		Dice		Jaccard	
	F1 Measure (%)					
Category	TF.IDF	WIDF	TF.IDF	WIDF	TF.IDF	WIDF
Baseball	95.08	96.77	91.80	95.24	91.80	95.24
Crypt	98.36	93.75	92.06	92.31	92.06	92.31
Electronics	83.02	83.64	83.64	85.19	83.64	85.19
ForSale	88.52	86.67	80.65	86.67	80.65	86.67
Graphics	90.00	93.10	88.52	89.66	88.52	89.66
Guns	95.24	91.80	94.92	91.80	94.92	91.80
Hockey	94.92	96.55	94.92	96.55	94.92	96.55
Misc.	93.55	90.32	86.67	91.80	86.67	91.80
Average (%)	**92.34**	**91.58**	**89.15**	**91.15**	**89.15**	**91.15**

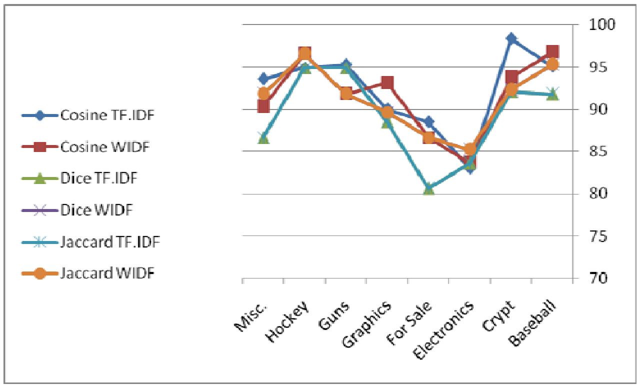

Fig. 2. F1 measure results

V. CONCLUSIONS

In this paper, we investigated different variations of VSM using KNN algorithm, these variations are: Cosine coefficient, Jacaard coefficient and Dice coefficient, using IDF and WIDF term weighting measures. The base of our comparisons is the F1 evaluation measure. The average F1 results indicated that Cosine based IDF outperformed the Cosine based WIDF, Dice based IDF, Dice based WIDF, Jaccard based IDF, and Jaccard based WIDF. We found out that there is consistency between Cosine based WIDF and Cosine based TF.IDF algorithms in which both of them outperformed Dice based TF.IDF, Dice based WIDF, Jaccard based TF.IDF, and Jaccard based WIDF. There are similarities between 1) Dice based TF.IDF and Jaccard based TF.IDF and 2) Dice based WIDF and Jaccard based WIDF in terms of F1 scores. We plan in near future to experiment other TC data collections especially Arabic data sets. Also we plan to propose a new TC technique based on association rule mining.

REFERENCES

[1] M. Antonie and O. Zaiane, "Text Document Categorization by Term Association," *Proceedings of the IEEE International Conference on Data Mining (ICDM '2002)*, pp.19-26, Maebashi City, Japan, December 9 - 12, 2002.

[2] F. Debole and F. Sebastiani, "Supervised term weighting for automated text categorization," *Proceedings of the 2003 ACM symposium on Applied computing*, pp. 784.788, ACM Press, 2003.

[3] Z. H. Deng, S. W. Tang, D. Q. Yang, M. Zhang, L. Y. Li and K. Q. Xie, "A comparative study on feature weight in text categorization," , pp. 588-597, 2004.

[4] T. Joachims, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features," *Proceedings of the European Conference on Machine Learning (ECML)*, pp. 173-142, Berlin, 1998, Springer.

[5] M. Lan, S. Y. Sung, H. B. Low and C. L. Tan, "A comparative study on term weighting schemes for text categorization," *Proceedings of the International Joint Conference on Neural Networks*, pp. 1032-1033, 2005.

[6] M. Lan, C. L. Tan and H. B. Low, "Proposing a New Term Weighting Scheme for Text Categorization," *Proceedings of the 21st National Conference on Artificial Intelligence*, pp. 763-768, July 2006.

[7] E. Leopold and J. Kindermann, "Text categorization with support vector machines. How to represent texts in input space?," *Machine Learning*, Vol. 46, No. 1-3, pp. 423-444, Jan. 2002.

[8] I. Moulinier, G. Raskinis and J. Ganascia, "Text categorization: a symbolic approach," *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval*, pp. 87-99, 1996.

[9] J. Quinlan "C4.5: Programs for machine learning," San Mateo, CA: Morgan Kaufmann, 1993.

[10] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGill-Hill, 1983.

[11] G. Salton, *Automatic Text Processing: The Transformation, Analysis Retrieval of Information by Computer*, Addison-Wesley, 1988.

[12] F.Sebastiani, "A Tutorial on Automated Text Categorisation, "*Proceedings of the ASAI-99, 1st Argentinian Symposium on Artificial Intelligence*, pp. 7-35, 1999.

[13] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, Vol. 34, No.1, pp. 1-47, March 2002.

[14] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, Vol.28, No.1, pp. 11-21, 1972.

[15] F. Thabtah, P. Cowling and Y. Peng, "MMAC: A new multi-class, multi-label associative classification approach," *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM '04)*, pp. 217-224, Brighton, UK, 2004.

[16] T. Tokunaga and M. Iwayama, "Text Categorization Based on Weighted Inverse Document Frequency," *Technical Report 94 TR0001, Department of Computer Science*, Tokyo Institute of Technology: Tokyo, Japan, 1994.

[17] K. Tzeras, S. Hartman, "Automatic indexing based on bayesian inference networks," *Proceedings of the 16th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR'93*), pp. 22-34, 1993.

[18] C. Van Rijsbergan, *Information retrieval*, Buttersmiths, London, 2nd Edition, 1979.

[19] E. Wiener, J. O. Pedersen, and A. S. Weigend, "A neural network approach to topic spotting," *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval* (*SDAIR'95*), pp. 317-332, Las Vegas, Nevada, 1995.

[20] Y. Yang and C. G. Chute, "An example-based mapping method for text categorization and retrieval," *ACM Transaction on Information Systems (TOIS)*, Vol. 12 No. 3, pp. 252-277, Jul. 1994.

[21] Y. Yang, "An evaluation of statistical approaches to text categorization," *Journal of Information Retrieval*, Vol.1 No. 1/2, pp. 67-88, May 1999.

[22] Y. Yang and X. Liu, "A re-examination of text categorization methods," *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval* (*SIGIR'99*), pp. 42-49, 1999.

[23] 20NewsGroups: http://people.csail.mit.edu/jrennie/20Newsgroups/