Journal of Engineering Science and Technology
Vol. 5, No. 1 (2010) 94 - 107

A NOVEL TERM WEIGHTING SCHEME MIDF FOR TEXT CATEGORIZATION

C. DEISY[1], M. GOWRI[2], S. BASKAR[3], S.M.A. KALAIARASI[4,*], N. RAMRAJ[5]

[1,2,3,5] Department of Computer Science and Engineering,
Thiagarajar college of Engineering, Madurai, Tamilnadu, India.
[4]Faculty of Information Science and Technoogy, Multimedia University,
Jalan Ayer Keroh Lama, Melaka, Malaysia.
*Corresponding Author: kalaiarasi@mmu.edu.my

Abstract

Text categorization is a task of automatically assigning documents to a set of predefined categories. Usually it involves a document representation method and term weighting scheme. This paper proposes a new term weighting scheme called Modified Inverse Document Frequency (MIDF) to improve the performance of text categorization. The document represented in MIDF is trained using the support vector machines classifier with radial basis function kernel. The experiments are carried out in Reuters-21578 corpora. The performance measures taken for text categorization are F_1–measure and cost measure. The proposed term weighting scheme performs better than the existing term weighting schemes.

Keywords: Text categorization, Support vector machine, Modified inverse document frequency, Text classification, Term weighting.

1. Introduction

Text Categorization (TC), also known as Topic Setting or Text Classification, is the task of automatically sorting a set of documents into categories (or classes, or topics) from a predefined set [1]. Automatic text categorization is treated as a supervised learning task. The goal of this task is to estimate a Boolean function to determine whether a given document belongs to the category or not by looking at the synonyms or prefix of that category. TC can be used in applications where there is a flow of dynamic information that needs to be organized. These applications include automated indexing of scientific articles according to the pr-

Nomenclatures

b	Constant
C	SVM misclassification tolerance parameter of the error function
$DF(i)$	The sum of the term frequency of the i^{th} document
$DFR(i)$	The document frequency (number of non-zero values for a document in each category)
fn	False negative
fp	False positive
$K(x, y)$	Kernel function
m	Total number of documents
N	Training cases
n	Total number of categories
p	Precision
$TF(i,j)$	Term frequency (number of times the word occur in the document)
r	Recall
tn	True negative
tp	True positive
w	Vector of coefficients
x_i	Independent variables
Y	Class labels

Greek symbols

ξ_i	Parameter for handling non-separable data (inputs)
Φ	Transform data from the input (independent) to the feature space

Abbreviations

IDF	Inverse Document Frequency
MIDF	Modified Inverse Document Frequency
RBF	Radial Basis Function
SVM	Support Vector Machines
TC	Text Categorization
TF	Term Frequency
WIDF	Weighted Inverse Document Frequency

defined 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, identification of document genre, authorship attribution, survey coding, and event automated essay grading [2].

The three different phases in the life cycle of a TC system are document indexing, classifier learning, and classifier evaluation. Document indexing is one of the most important issues in TC, which includes document representation and a term weighting scheme. For document representation, bag-of-words [3] is the most common way to represent the context of texts. The advantage of this approach is simplicity as only the frequency of word in a document is recorded. Here, for all the predefined categories, the synonyms and prefix words for the category are found and it helps to assign any document to that category based on the synonym or prefix of a term.

Term Frequency (TF) is the simplest measure to weight each term in a text. The drawback of TF is the difficulty in finding the optimal thresholds [4]. Also TF is known to improve recall but does not improve precision. While TF concerns term occurrence within a text, Inverse Document Frequency (IDF) concerns term occurrence across a collection of texts. The intuitive meaning of IDF is that terms, which rarely occur in a collection of texts, are valuable which improve precision. Thus, the combination of TF and IDF improves recall and precision respectively that gives better performance. All TC researchers use only the product of TF and IDF. A drawback of IDF is that all texts that contain a certain term are treated equally i.e., IDF does not distinguish between one occurrence of a term in a text and many [4]. The drawback of TF.IDF is that when a new document occurs, recalculation of weighting factors to all documents is needed since it depends on number of documents. Weighted Inverse Document Frequency (WIDF) [4] overcomes this by weighting term that sums up to one over the collection of texts. WIDF itself improves both the precision and recall. A drawback of WIDF is that when the number of documents becomes large, the terms that have the nearest frequency, have almost equal weight, which makes the learning task more difficult.

Hongzhi Xu and Chunping Li proposed a term weighting scheme TM2 [5] for text categorization. They tried to explore the information from term distribution among different categories. The Reuter-21578 corpora and top six categories are chosen for their evaluation. They investigated that the term weighting scheme TM2, not only consider the frequency of the documents that contains the term t , but also consider the number of times that t occurs in different documents of different categories. For this, they fixed the number of features to 2000 and randomly select 4, 8, 12, 16, 20 categories for 5 times respectively for their implementation. Finally they found that the performance of TM2 is significantly affected when the number of categories increased [5].

To overcome this, a new method Modified Inverse Document Frequency (MIDF) is proposed in this paper. This scheme depends on term frequency and document frequency but not the number of documents in the collection. The proposed term weighting scheme is a normalized term frequency over the collection, which provides the correct terms for learning.

The different classification learning methods applied for TC are Naïve Bayes [6], Rocchio's Algorithm [7], J4.8 Decision tree/rule learner [8, 9], Perceptrons [10], Nearest Neighbor [10, 11] and Support Vector Machines (SVM) [1, 12, 13]. In all the above learning methods TF.IDF method is used for document indexing. Recently, SVM methods for Text Categorization have attracted more attention, since they are the most accurate text classifiers. The distinguishing feature of SVMs is the use of a subset of training examples. In SVM, only the support vectors are used to build the classifier.

In this paper, to improve the performance of SVM based TC, a novel term weighting scheme MIDF have been proposed. The classifier evaluation is provided by F_1-measure and categorization cost. For categorization Reuters-21578 [14] corpora has been chosen.

The remainder of the paper is organized as follows. Section 2 presents Text Categorization and Section 3, describes the SVM and different kernel functions. Section 4 discusses the proposed term weighting scheme for text categorization

method. In section 5, results and discussion are given followed by conclusions in section 6.

2. Text Categorization

The problem of Text Categorization can be described as the classification of documents into multiple categories. A set of n categories $\{C_1, C_2, \ldots, C_n\}$ to which m documents $\{D_1, D_2, \ldots, D_m\}$ are assigned. Figure 1 shows the categorization problem [2].

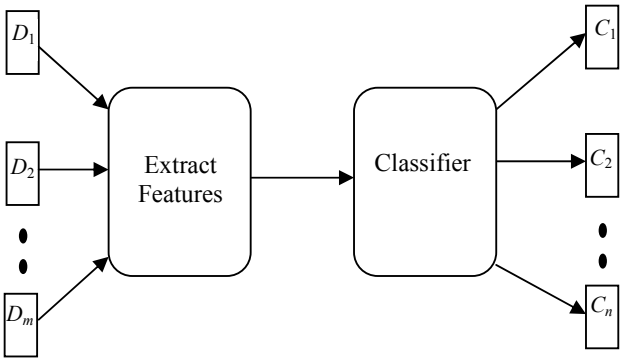

Fig. 1. Assignment of Documents to Categories.

The n categories are predefined with specific keywords that differentiate any category C_i from every other category C_j. The process of identifying these keywords is called feature extraction. TC is a subjective task i.e. when experts (human or artificial) decide to classify document D_i under category C_j. The decision depends on the subjective judgment of the expert. Using machine-learning techniques, the objective is to learn classifiers from text samples, which assign categories automatically. To facilitate effective and efficient learning, each category is treated as a separate binary classification problem, which expresses that a document should be assigned to a particular category, or not.

3. Support Vector Machines (SVM)

The basic idea of Support Vector Machine [11, 12] is to find an optimal hyper plane to separate two classes with the largest margin from pre-classified data [16]. The use of the maximum-margin hyper plane is motivated by Vapnik Chervonenkis theory [11, 12], which provides a probabilistic test error bound that is minimized when the margin is maximized [12].

To construct an optimal hyper plane, SVM uses an iterative training algorithm, which is used to minimize an error function. SVM used here is C-SVM. For C-SVM, training involves the minimization of the error function given in the following equation

$$\frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i \tag{1}$$

Subject to the constraints given in Eq. (2)

$$y_i\left(w^T \Phi(x_i) + b\right) \geq 1 - \xi_i \text{ and } \xi_i \geq 0, \; i = 1,\ldots\ldots, N \tag{2}$$

where C is the capacity constant or penalty parameter of the error function, w is the vector of coefficients, b is a constant and ξ_i is parameter for handling non-separable data (inputs). The index i is the labels and N is the training cases. Note that $y \in \pm 1$ is the class labels and x_i is the independent variables. The kernel Φ is used to transform data from the input (independent) to the feature space [13].

In SVM there are four common kernels:

- linear
- polynomial
- Radial Basis Function (RBF)
- sigmoid

There are many other kernels apart from the common kernels. In general RBF is a reasonable first choice because the kernel matrix using sigmoid may not be positive definite and in general its accuracy is not better than RBF [17], linear is a special case of RBF, and polynomial may have numerical difficulties if a high degree is used. A comparison of linear, polynomial and RBF kernels are made.

The linear kernel function is given by,

$$k(x, y) = (x \,.\, y) \tag{3}$$

The polynomial kernel function with dimension d is given by,

$$k(x, y) = (x \,.\, y)^d \tag{4}$$

The sigmoid kernel function with gain and offset Θ is given by,

$$k(x, y) = \tanh\left(K(x, y) + \theta\right) \tag{5}$$

For RBF, the kernel function is defined in Eq. (6)

$$k(x, y) = \exp\left(- \|x - y\|^2 / (2\sigma^{2})\right) \tag{6}$$

A linear classifier example is shown in Fig. 2. In this example a classifier that separates a set of objects into their respective groups (Circle and Square in this case) with a line.

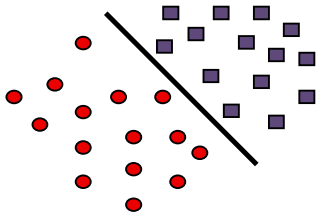

Fig. 2. Linear Classifier.

But most classification tasks are not that simple which is illustrated in Fig. 3.

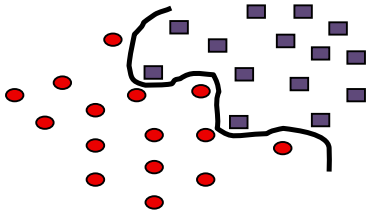

Fig. 3. Complicated Task Classified by a Curve.

In this case it is clear that a curve is needed to fully separate the circle and square objects. The illustration in Fig. 4 shows the basic idea behind SVM kernels. The original objects are mapped, i.e., rearranged, using a set of mathematical functions known as kernels. The process of rearranging the objects is known as mapping (transformation) [13]. Although the transformation may be non-linear and the transformed space is high dimensional, as the result of kernels' transformation, the mapped objects is linearly separable and, thus, instead of constructing the complex curve.

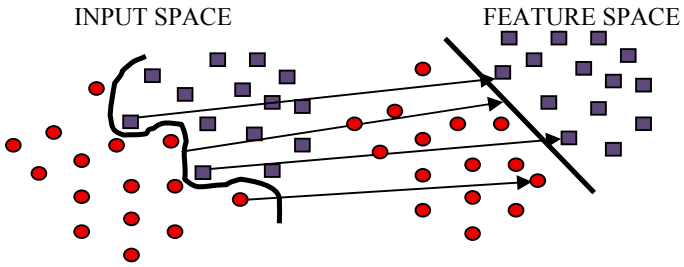

Fig. 4. Kernel Mapping.

There are many existing SVM library, such as rainbow [18], LIBSVM [16, 18] etc. LIBSVM is an integrated software for support vector classification, (C-SVC, nu-SVC), regression (epsilon-SVR, nu-SVR) etc [19]. The SVM code used for the experiments is the work of Anton Schwaighofer [20].

4. Text Categorization Based on MIDF

The SVM method has been introduced in TC by Joachims [6] and subsequently used in TC works [1, 2, 5-7]. One advantage that SVMs offer for TC is that dimensionality reduction is usually not needed, as SVMs tend to be fairly robust to over fitting and can scale up to considerable dimensionalities. The feature selection tends to be detrimental to the performance of SVM [1]. In the existing methods of TC using SVM, bag-of-words is used for document representation. In this, semantic

relationships between words are not taken and when a new document with a new word is to be categorized then bag-of-words [3] should be altered which requires a rebuild the representation for all documents. To overcome this, new document representation method category-of-words has been proposed. Also TF.IDF is the commonly used term weighting scheme in which, when a new document occurs needs a recalculation of weighting factor for all documents since it depends on number of documents. To overcome this, MIDF is proposed. The architecture of proposed Text Categorization method is shown in Fig. 5.

The basic design of this work is to transform documents into a representation suitable for categorization and then categorize documents to the predefined categories based on the training weights.

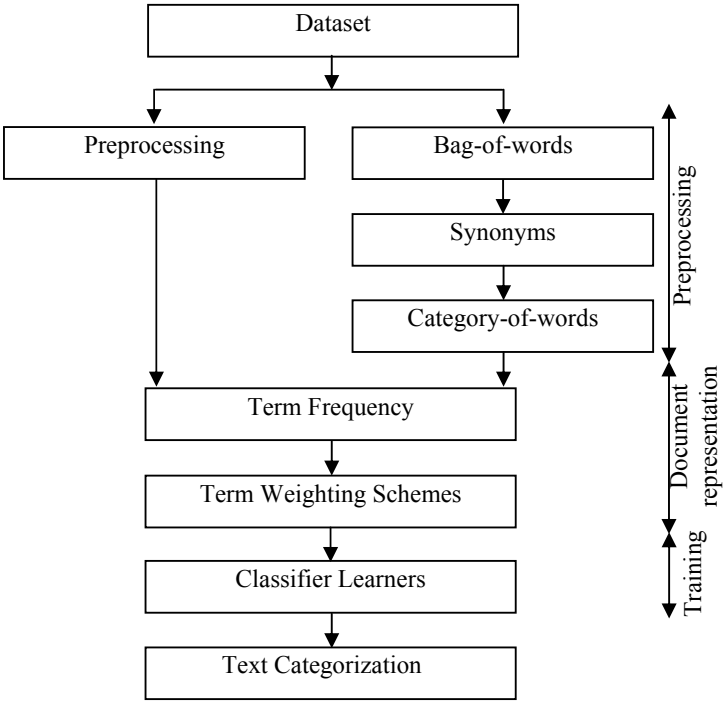

Fig. 5. Architecture of Proposed Text Categorization Method.

4.1. Preprocessing phase

In the preprocessing phase shown in Fig. 5, feature reduction is performed. The two kinds of feature reduction are removal of stop words [21] as they are useless for classification and stemming [3] that involves mapping words with the same meaning to once morphology. The Porter Stemming algorithm [22] is used for this purpose. This algorithm strips common terminating strings (suffixes) from words in order to reduce them to their roots or stems. Each word undergoes 6 steps to reduce to their stem. A list of suffices to be removed is specified together

with some conditions (for instance, the minimum length of the remaining stem). When the conditions are met, the suffix is stripped or replaced by another suffix.

The dataset is also taken to find the bag-of-words [3], i.e., the unique words excluding stop words among the total documents in the dataset. Then the synonyms for the bag-of-words [3] are found using Word Net [15]. For each category, the related words such as synonyms and prefix words are found which is named as category-of-words. For the predefined categories, the category-of-words are found which helps to assign any document to that category based on the synonym or prefix of a term.

4.2. Document representation phase

The main function of document representation phase is to convert terms which are strings to feature IDs which are integers of greater than or equal to 0. For this, computes Term Frequency (TF). The TF is the count of category-of-words of every category in each document. So the documents term frequency for a category is the occurrence of the prefix and synonym words of that category. This is represented in Fig. 6, as a two-dimensional vector space.

$$\begin{array}{cccccc} & [C_1 & C_2 & ..... & C_n] \\ D_1 & TF(1,1) & TF(1,2) & ..... & TF(1,n) \\ D_2 & TF(2,1) & TF(2,2) & ..... & TF(2,n) \\ . & . & . & & . \\ . & . & . & & . \\ . & . & . & & . \\ D_m & TF(m,1) & TF(m,2) & ..... & TF(m,n) \end{array}$$

Fig. 6. Document Representation.

Also these feature values TF(i,j) are called weights. Different term weighting factors such as TF.IDF [4], WIDF [4] exists which makes training efficient when compared to term frequency. In most classifiers, the standard TF.IDF function is used, which is defined by Salton and Buckley in 1988 [4], given in Eq. (7)

$$\text{TF.IDF}(i,j) = TF(i,j).\log 2\left(\frac{|m|}{DFR(i)}\right) \tag{7}$$

Here $TF(i, j)$ is the term frequency (number of times the word occur in the document) of the i^{th} document and the j^{th} category, $DFR(i)$ is the document frequency (number of non-zero values for a document in each category) of the i^{th} document and m is the total number of documents.

Similarly Weighted Inverse Document Frequency (WIDF) [4] is defined in Eq. (8),

$$\text{WIDF}(i,j) = \frac{TF(i,j)}{DF(i)} \tag{8}$$

where the document frequency $DF(i)$ is the sum of the term frequency of the i^{th} document.

The new proposed Modified Inverse Document Frequency (MIDF) is defined in Eq. (9),

$$\mathrm{MIDF}(i,j) = \left[1 + \log 2\{TF(i,j)\}\right] \frac{DFR(i)}{DF(i)} \tag{9}$$

where the document frequency $DF(i)$ is the sum of the term frequency and $DFR(i,j)$ is the number of non-zero values of the i^{th} document.

MIDF performs better than TF.IDF and WIDF which is proved theoretically. The $\log_2(TF(i,j))$ is taken because SVM is a binary classifier that works well with this type of term frequency. If $TF(i,j)$ is one, then the MIDF(i,j) becomes zero. To overcome this one is added to $\log_2(TF(i,j))$.

$DFR(i)/DF(i)$ is taken since it normalizes the value between 0 to 1. The entire formulae automatically normalize the value for the provided m documents.

Also TF.IDF is dependent on the number of documents. It works very well for TC with fixed documents but in real time applications such as automated indexing, filing patents [2], where the number of documents increases over time. So when a new document occurs, the TF.IDF should be recalculated for all the documents and network should be retrained which leads to higher computation time. Hence TF.IDF is not suitable for real time TC. To improve the performance and reduce the online computation burden a new term weighting scheme, MIDF is proposed.

4.3. Training phase

After transformation of these documents into a representation suitable for training, the training is undertaken. For training, two-third of the documents of the dataset is considered. The remaining one-third of documents is used for testing. The support vector machines (SVM) [16] is a binary classifier. Text Categorization is a multi-class problem. For each category, a binary SVM classifier is designed. A document may be assigned to more than one category. So, for the text categorization, the SVM classifiers are needed for the number of predefined categories.

The training set is provided to the SVM for training. The input to the SVM is a set of N pairs of training documents and categories, $\{(x_{11}, y_1),\ldots, (x_{mn}, y_n)\}$. Each x_{ij} is the MIDF weight value of document i and category j. Each y_i is either +1 (belongs to the category) or -1. The input vector provided is converted to the feature vector with the help of the linear, polynomial or RBF kernels and the training is performed. The intuition behind the SVM training is to find the support vectors and bias for each category.

4.4. Testing phase

In the testing phase with the trained weights the new document or set of documents are categorized. The documents are preprocessed and are represented

in the proposed term weighting scheme MIDF. After the representation the documents are categorized based upon the training weights of the classifiers.

Here the SVM function accepts an unseen document x_i and returns +1 or -1. The test vector x_i is represented in the MIDF term weighting scheme and transformed from input space into feature space using the kernel Φ (here linear, polynomial and RBF are used). The support vectors $x_1, x_2, \ldots x_n$ are also similarly transformed. The dot product of x with n support vectors is computed and weighted using $v_i s(\alpha_i y_i)$ found by applying QP algorithm. The sign of the sum of the dot product and the bias decides the classification of the document to a category. If the sign is positive, then the document can be categorized to that category. The same procedure is repeated for all other categories. The new classification process for the unseen document to a category is described in the Fig. 7.

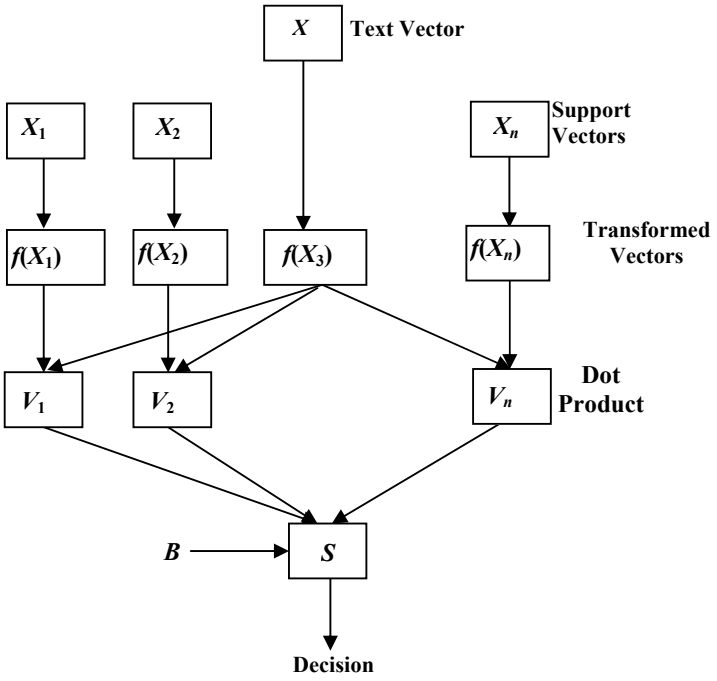

Fig. 7. Classification of an Unseen Document.

5. Experiments and Results

5.1. Corpora

The Reuters-21578 corpora [14] contain 12902 documents and 118 categories. The documents are news-stories collected by David Lewis in 1987. This dataset is preprocessed and the category-of-words is found. Then document are represented

in MIDF term weighting scheme. For training two-third documents and for testing one-third documents are taken.

5.2. Training and testing

For training 8602 documents are taken in random from the collection and the remaining 4300 documents are taken for testing. Here, 118 binary classifiers are built since there are 118 categories in the corpora. Each binary classifier holds the Lagrangian multipliers alpha and a bias value. With this, the new document is categorized. The average support vectors during RBF training using MIDF is 35.

Anton Schwaighofer [20] code for SVM is chosen because of the simplicity of programming, flexibility to provide values for the variables, different SVM formulations and Mat lab sources [23]. It provides a simple interface where users can easily link it with users' own programs.

5.3. Performance metrics

The performance of SVM with RBF kernel based TC is compared for various term weighting schemes based on F_1-measure and cost measure. The F_1-measure [17, 24] is the calculation accuracy for the categorization of the text. The micro average for the top ten categories and the macro average for the 118 categories are provided. The cost measure is the misclassification rate of the documents [24]. The performance measures are explained in the following section.

5.3.1. F_1-Measure

F_1-Measure is given in Eq.(10)

$$F_1 = \frac{2rp}{r+p} \tag{10}$$

where r is recall which is given by Eq. (11),

$$r = \frac{\text{total number of correctly classified documents}}{\text{total number of correct documents}}$$

$$r = \frac{tp}{tp+fn} \tag{11}$$

and p is the precision which is given by Eq. (12),

$$p = \frac{\text{total number of correctly classified documents}}{\text{total number of documents classified}}$$

$$p = \frac{tp}{tp+fp} \tag{12}$$

5.3.2. Cost Measure

The categorization cost [24] is also calculated which is given by Eq. (13),

$$Cost = Miss + False\_Alarm \tag{13}$$

where miss is given by

$$Miss = \frac{\text{total number of incorrectly documents that are classified}}{\text{total number of correct documents}}$$

$$Miss = \frac{fn}{tp + fn} \tag{14}$$

and *False_ Alarm* is given by Eq. (15),

$$False\_Alarm = \frac{\text{total number of correct documents that are misclassified}}{\text{total number of incorrect documents}}$$

$$False\_Alarm = t\frac{fp}{tn + fp} \tag{15}$$

Table 1. Performance Metrics of SVM with RBF Kernel.

Category	F_1-Measure			Cost Measure		
	TF.IDF	**WIDF**	**MIDF**	**TF.IDF**	**WIDF**	**MIDF**
acq	94.041	99.789	**99.795**	11.247	0.421	**0.409**
corn	69.048	87.180	**98.148**	47.273	22.727	**3.636**
crude	84.158	91.163	**99.138**	27.350	16.239	**1.709**
earn	82.160	94.958	**99.598**	30.025	9.600	**0.800**
grain	78.457	93.923	**98.660**	35.450	10.126	**2.646**
interest	69.194	89.600	**98.901**	47.101	18.841	**2.174**
money-fx	92.173	98.522	**99.864**	14.430	1.715	**0.272**
ship	76.071	92.000	**99.589**	38.164	14.815	**0.820**
trade	90.938	99.137	**99.853**	15.769	0.949	**0.293**
wheat	71.963	87.243	**98.519**	43.796	22.628	**2.920**
Macro Avg. / Total Cost	63.040	75.596	**88.351**	47.025	30.922	**14.720**

From Table 1, it is clear that SVM with RBF kernel based on MIDF performs better than TF.IDF and WIDF. Also a comparison with linear and polynomial kernel is performed where RBF kernel based on MIDF works better.

From Fig. 8, the F_1-Measure for SVM with RBF kernel based on MIDF performs well when compared to TF.IDF and WIDF. The micro-average for the top ten categories and macro-average of the 118 categories are provided. The overall performance of MIDF is better when compared to TF.IDF and WIDF. Also when compare with linear kernel, RBF performs well.

From Fig. 9, the Cost Measure for SVM with RBF kernel based on MIDF performs well when compared to TF.IDF and WIDF. The total cost of the 118 categories is also good. The overall performance of MIDF is better when

compared to TF.IDF and WIDF. Also when compared with linear kernel, RBF performs well.

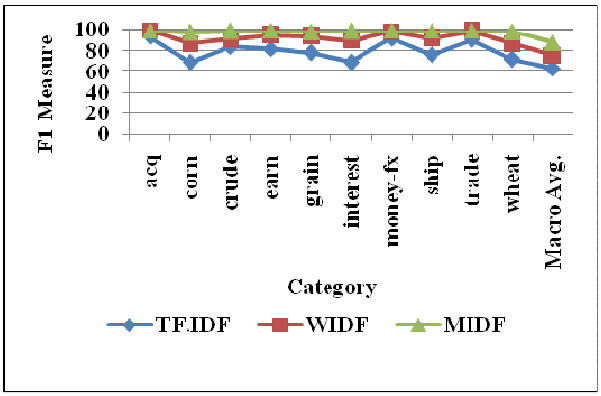

Fig. 8. F1-Measure for SVM with RBF Kernel.

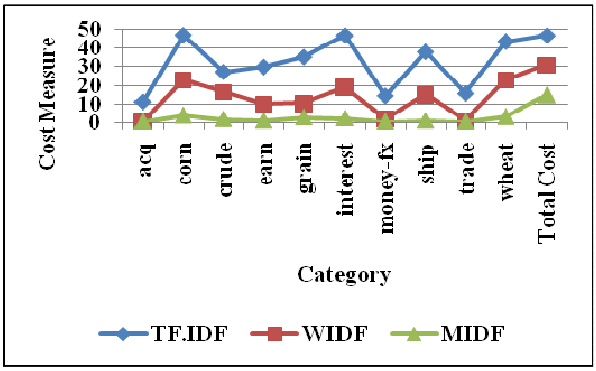

Fig. 9. Cost Measure for SVM with RBF Kernel.

6. Conclusions

In this paper, to improve the performance of SVM based text categorization (TC) a new term weighting scheme called Modified Inverse Document Frequency (MIDF) is proposed. The experiments are conducted for Reuters 21578 corpora, 12902 documents and 118 categories. The performance of TF.IDF, WIDF, and MIDF based SVM -TC (linear, RBF kernel) schemes are compared with TF.IDF, WIDF, MIDF based text categorization. The performances of TC schemes are compared with respect to micro averaged F_1-measure, macro-averaged F_1-measure and categorization cost for the top ten categories. The results show that the overall performance of MIDF based SVM -TC with RBF kernel performs better than the corresponding TF.IDF and WIDF based approaches and linear kernel. As the online computation time of MIDF method is inherently less, MIDF based TC scheme can be recommended for applications where there is a large flow of dynamic information that needs to be organized.

References

1. Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), 1-47.
2. Konchady, M. (2006). *Text mining application programming*. India Edison, Thomson Learning.
3. Salton, G.; and McGill, M.J. (1983). *Introduction to modern information retrieval*. McGraw-Hill, New York.
4. Tokunga (1994). Text categorization based on weighted inverse document frequency. *Tech. Rep.TR0001*, Department of Computer Science, Tokyo Institute of Technology, Tokyo Japan.
5. Xu, H.; and Li, C. (2007). A Novel term weighting scheme for automated text Categorization. *Proceedings of Seventh International Conference on Intelligent Systems Design and Applications*, Rio de Janeiro.
6. Joachims, T. (1997). A probalistic analysis of the Rocchio algorithm with tfidf for text categorization. *Proceedings of International Conference on Machine Learning (ICML*), Nashville, TN, USA.
7. Rocchio, J. (1971). *Relevance feedback in information retrieval*, In G. Salton, editor, The SMART Retrieval System: Experiments in Automatic Document Processing. USA, Prentice-Hall In., Englewood Cliffs.
8. Ian Witten. I; and Eibe Frank (2005). *Data mining: practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.
9. Quinlan, J.R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann.
10. Mitchell, T.M. (2006). *Machine learning*, McGraw-Hill.
11. Yang, Y. (1997). An evaluation of statistical approaches to text categorization. *Tech. Rep. CMUCS, Carnegie,* Mellon University, 97-127.
12. Support Vector Machine: http://en.wikipedia.org/wiki/Support_vector_machine.
13. Support Vector Machines (SVM): http://www.statsoft.com/textbook/stsvm.html.
14. Reuters 21578 corpus: http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.tar.gz.
15. Word net 2.3 for Windows tool: http://wordnet.princeton.edu/2.1/WordNet-2.1.exe.
16. Statement of Work: http://128.82.7.230/categorization/DTICCategorization.doc.
17. LIBSVM FAQ: http://www.csie.ntu.edu.tw/~cjlin/libsvm/faq.html.
18. Makato Suzuki; and Shigeichi Hirasawa. (2007). Text categorization based in the ratio of word frequency in each categories. *Proceedings of International Joint Conference on Neural Networks*, Oriando, Florida, USA, 12-17.
19. LibSVM site: http://www.csie.ntu.edu.tw/~cjlin/libsvm/.
20. Anton Schwaighofer code for SVM:http://ide.first.fraunhofer.de/~anton/software.html.
21. Stop words: http://dev.mysql.com/doc/refman/5.0/en/fulltextstopwords.html.
22. Porter, M.F. (1980). An algorithm for suffix stripping. *Program Automated Library and Information Systems*, 14(3), 130-137.
23. Mat lab: http://www.mathworks.com/.
24. Tzu-Chuan Chou; and Meng Chang Chen (2008).Using incremental PLSI for threshold resilient on-line event analysis. *IEEE Trans. Knowledge and Data Engineering*, 20(3), 289-299.