Open access • Book Chapter • DOI:10.1007/3-540-45357-1_9

Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification

— Source link

Eui-Hong Han, George Karypis, Vipin Kumar

Institutions: University of Minnesota

Published on: 16 Apr 2001 - Pacific-Asia Conference on Knowledge Discovery and Data Mining

Topics: Statistical classification, k-nearest neighbors algorithm, Feature (machine learning) and Categorization

Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 99-019

Text Categorization Using Weight Adjusted k-Nearest Neighbor Classification

Euihong (sam) Han, George Karypis, and Vipin Kumar

May 17, 1999

Text Categorization Using Weight Adjusted *k*-Nearest Neighbor Classification [*]

Eui-Hong (Sam) Han George Karypis, Vipin Kumar

Department of Computer Science and Engineering
University of Minnesota
4-192 EECS Bldg., 200 Union St. SE
Minneapolis, MN 55455, USA

{han,karypis,kumar}@cs.umn.edu

Last updated on March 20, 1999 at 12:11am

Abstract

Categorization of documents is challenging, as the number of discriminating words can be very large. We present a nearest neighbor classification scheme for text categorization in which the importance of discriminating words is learned using mutual information and weight adjustment techniques. The nearest neighbors for a particular document are then computed based on the matching words and their weights. We evaluate our scheme on both synthetic and real world documents. Our experiments with synthetic data sets show that this scheme is robust under different emulated conditions. Empirical results on real world documents demonstrate that this scheme outperforms state of the art classification algorithms such as C4.5, RIPPER, Rainbow, and PEBLS.

Keywords: Classification, text categorization, nearest neighbor classification, weight adjustment, data mining.

[*] This work was supported by NSF grant ASC-9634719, Army Research Office contract DA/DAAH04-95-1-0538, Cray Research Inc. Fellowship, and IBM partnership award, the content of which does not necessarily reflect the policy of the government, and no official endorsement should be inferred. Access to computing facilities was provided by AHPCRC, Minnesota Supercomputer Institute, Cray Research Inc., and NSF grant CDA-9414015. See **http://www.cs.umn.edu/~han** for other related papers.

1 Introduction

Text categorization is the task of deciding whether a document belongs to a set of prespecified classes of documents. The amount of online document data is growing very fast. For instance, the World Wide Web is a vast resource of information and services that continues to grow rapidly. Another examples can be found in the growing size of digital library, hospitals and other work places where the diagnosis and field reports are readily available online.

The task of assigning these documents to a set of prespecified classes of documents is challenging due to the large amount of documents in these fields. Automatic classification schemes greatly enhance the process of categorization. For instance, Yahoo! [Yah99] currently uses human experts to categorize the documents. However, the way WWW documents are growing, this would become more difficult. At West Group, human indexers read legal documents and index them manually [CT97]. This step becomes a bottleneck in publishing legal documents for the legal community. Web browser users keep bookmarks to remember sites they are interested in. Often times, they categorize these sites according to their interests such as business, sports, travel, books, and movies. It will be a great help to the users if the automatic text categorization could classify all the searched documents from the web based on the existing bookmarked documents of different user categories.

A major difficulty of applying existing classification algorithms in text categorization domain is the high dimensionality nature of document data sets. The words occur in the document sets become variables or attributes for the classification problem. A relatively moderate size of document sets could easily have number of distinct words in tens of thousands. Many existing algorithms simply would not work with these many number of attributes. Several feature selection methods based on document frequency, mutual information, or information gain could be used to reduce the number of words [YP97, Joa97, McC96]. However, if we become too aggressive in reducing the number of words, then we might lose critical information for categorization tasks. Normally, the number of words after feature selection could be still in thousands.

In addition to the large number of words, text categorization pauses more problems as the words are tend to be dependent. A lot of times, single words do not characterize categories of documents, but a pair or group of words determine the categories. In other word, based on the occurrence of a single word in the document, the document cannot be categorized to a certain class. The same word could occur in documents of one category and also in documents of the other categories. However, this kind of word cannot be simply removed from the data set using feature selection, as this word combined with other words could uniquely determine right categories of the documents.

There are several classification schemes that can be potentially used for text categorization. However, many of these existing schemes do not work well in the text categorization task due to the problems mentioned above. For example, widely used classification decision tree induction algorithm like C4.5 [Qui93] or rule induction algorithms such as C4.5rules [Qui93] and RIPPER [Coh95] do not work well with large number of attributes. Even though Naive-Bayes classification techniques, such as Rainbow [McC96], are popular in text categorization [LG94, LR94, Lew98, MN98], they have a major limitation due to the independence assumption they make while words in document data sets tend to be dependent.

k-nearest neighbor (k-NN) classification is an instance-based learning algorithm that has shown to be very effective in text classification [Yan94, CH98]. The success of this scheme is due to the availability of effective similarity measures such as *cosine* measure [Sal89]. However, the effectiveness of these similarity measures become worse as the number of words increases. PEBLS [CS93] is a k-NN classification algorithm that incorporates class information in the similarity measure. However, the similarity measure is developed mainly for the data sets with categorical attributes. Even though, document data sets can be regarded as categorical data by considering each word to be either present or absent in a document, the effectiveness of the similarity measure of PEBLS in the text classification task is questionable.

In this paper, we propose Weight Adjusted k-Nearest Neighbor (WAKNN) classification algorithm that is based on the k-NN classification paradigm. In WAKNN, the importance of each word in the classification of a training document set is learned and the weight vector reflecting this importance is maintained. The weight vector is used in the similarity measure computation such that important words contribute more in the similarity measure. Experiments on several synthetic and real life data sets show the promise of WAKNN, as it outperformed other classifiers in terms of classification accuracies.

2 Weight Adjusted *k*-Nearest Neighbor Classification Algorithm (WAKNN)

In this section, we present the details of WAKNN. The key aspects of WAKNN are how to initialize the weight vector, how to find the best possible weight vector, and how to use the weight vector to define a better similarity measure. Here are the major steps of WAKNN:

1. Construct training matrix, D, where each row correspond to a training document, each column represent a word, and value in the matrix $D(i, j)$ corresponds to the number of occurrences of word j in document i.
2. Normalize word frequencies in each document such that they add up to 1.0. This step normalizes the difference in document lengths.
3. Find mutual information of each word using Equation 2 described later and initialize weight vector W with these values.
4. Determine k nearest neighbors for each training document using the weighted cosine similarity measure of Equation 1 described later with this initial weight vector.
5. Calculate the goodness of this initial weight vector using an objective function, $Obj(D, W, p)$, with parameter p described later.
6. While there is an improvement in $Obj(D, W, p)$, repeat the following steps:
   (a) For each word i, determine the value of W_i that gives the best $Obj(D, W, p)$. New possible values for W_i are proposed by multiplying the original W_i with different multiplication factors.
   (b) Select a word j that gives the best overall $Obj(D, W, p)$ from the previous step, and update W_j with this new value.

In classifying a test document, we first construct a test vector according to the steps 1 and 2 of WAKNN. We find k nearest neighbors of the test document from the training documents using the weighted cosine similarity measure with the weight learned from WAKNN. We then sum up the similarities to the k neighbors according to their class labels. We classify the test document according to the class with the most similarity sum.

Weighted Cosine Similarity Measure We have followed vector space model commonly used in Information Retrieval systems [Sal89]. In this vector model, each document is a vector and its element corresponds to words in the whole document set. The whole training set can be viewed as a matrix where each row is a document and its columns are words. The values in the matrix can be binary, 1 for presence of the word and 0 for absence of the word. They can also be the within-document word frequency (TF), Inverse Document Frequency (IDF), or TFIDF which is the combination of TF and IDF [Sal89].

Most popularly used schemes in Information Retrieval is TFIDF. However, previous studies in text classification [YC94] and clustering [BGG$^+$99] indicate that TFIDF is not very effective. In WAKNN, we have adopted TF as the entry in the matrix and normalized per row such that each row adds up to 1.0. This step eliminates problems due to the differences in the document size.

For the similarity between documents, cosine similarity measure is commonly used [Sal89]. We define a weighted cosine measure between document X and Y with weight vector W and set of terms (or words) T as

$$cos(X, Y, W) = \frac{\sum_{t \in T} (X_t \times W_t) \times (Y_t \times W_t)}{\sqrt{\sum_{t \in T} (X_t \times W_t)^2} \times \sqrt{\sum_{t \in T} (Y_t \times W_t)^2}}, \tag{1}$$

where X_t and Y_t are normalized TF of word t for X and Y, respectively, and W_t is the weight of word t.

Weight Initialization Using Mutual Information Mutual information of each word with the class variable has been used in Information Retrieval domain to select important features [CT91, YP97]. Mutual information of a word captures the amount of information gained in classifying documents by knowing the presence and absence of the word. The mutual information of a word w with respect to classes C is defined as

$$MI(w) = \sum_{c \in C} \left(P(c, w) \log \frac{P(c, w)}{P(c)P(w)} + P(c, \overline{w}) \log \frac{P(c, \overline{w})}{P(c)P(\overline{w})} \right) \tag{2}$$

where $P(c)$ is the probability of class c, $P(w)$ is the probability of the presence of word w, and $P(\overline{w})$ is the probability of the absence of word w, and $P(c, w)$, $P(c, w)$, and $P(c, \overline{w})$ are joint probabilities. Note that when a word has high mutual information value, it provides more information in the classification task. Note that mutual information captures the importance of each word separately. It does not consider the dependence among different words. For instance, when a word does not discriminate among classes by itself, but does so with other words, the mutual information of this word is very small. Even with this limitation, mutual information provides a good starting point for the weight adjustment. In WAKNN, we use the mutual information of each word to initialize the weight vector.

Weight Adjustment Based on Objective Functions In the weight adjustment step, we are trying to find the optimal weight vector for the classification task at hand. Starting with the weight vector obtained using the mutual information, we try to make a small change to the weight vector to see if we improve the objective function related to the classification.

There are two important issues in this step. The first issue is how to evaluate the weight change. In the paradigm of k nearest neighbor classification, the objective function is closely related to class labels of neighbors of each training document. The best case would be when all the neighbors of each training document have the same class labels as the training document. The objective function of WAKNN is defined as:

$$\text{Obj}(D, W, p) = |\{d | d \in D \text{ and Correct}(d, D, W, p)\}|$$

where D is the training document matrix, W is the weight vector, and predicate Correct(d, D, W, p) is true if out of k nearest neighbors of d from D calculated using the weighted cosine measure, the majority neighbors are from the same class as d and the sum of the similarities to these majority neighbors are at least p percent of the total k neighbor similarity sum. We call p as the majority percentage. The majority percentage prevents a training document to be considered to be correctly classified when it does not have many neighbors of the same class label. For instance, when $k = 5$, $p = 0$, and the number of classes is 5, if one training document has one neighbor from each of these classes, then this training document is considered to be correctly classified if the similarity to one neighbor with the same class is the highest among other 4 neighbors. On the other hand, in the same situation with $p = 50$, this training sample will be considered to be correctly classified only if the similarity of the training document to this particular neighbor is at least 50% of the total similarity sum. This guarantees that this training document is considered to be correctly classified only if similarity to other 4 neighbors are very weak.

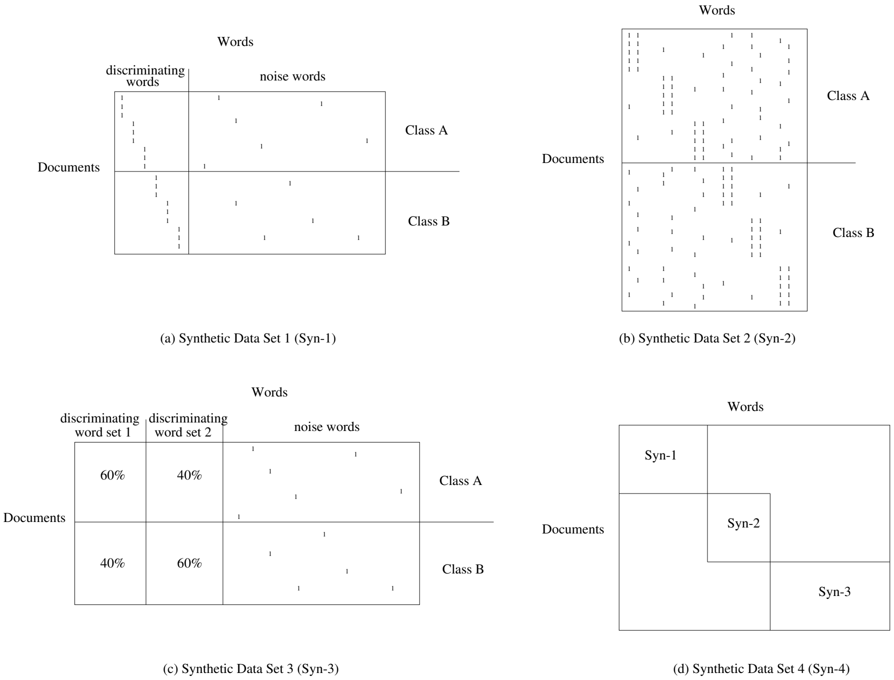
The second issue is how to propose possible changes to the weight. For the proposal of the possible changes, we adopted to change one word weight at a time. For each word weight, we multiply the current word weight with different multiplication factors (e.g., {0.2, 0.8, 1.5, 2.0, 4.0}). For each of these changes, we evaluate if this change improves the objective function. We remember the best change for each word. We pick the best word and its weight that gives the best value according to the objective function. We update the weight vector with this value and continue to the next round.

3 Experimental Results

In this section, we compare kNN-mut which uses the weight vector obtained using mutual information as the final weight vector and WAKNN against kNN, C4.5 [Qui93], RIPPER [Coh95], PEBLS [CS93], Rainbow [McC96], VSM [Low95] on several synthetic and real data sets. VSM is another k-NN scheme in which the weight of the attributes is learned through conjugate gradient optimization. There are two parameters of importance in WAKNN. The first is the choice of number of neighbors (k) and the second is the majority percentage in the objective function discussed in Section 2. We have performed experiments of changing k from 1 to 50 and majority percentage from 0% (i.e., simple majority) to 90%. The results (not reported here due to the space constraint) show that the classification accuracy do not vary significantly when k is between 5 and 30, and when the majority percentage is from 30% to 70%. In the subsequent experiments, we have chosen k to be 10 and the majority percentage to be 50%. For C4.5 and RIPPER, we had a choice of either regarding each word as a discrete variable or continuous variable. The results reported here are obtained by regarding each word as a discrete variable. The results from experiments where each word is regarded as a continuous variable did not differ from the reported results. For PEBLS, there is an option to weight instances differently. We tried this option, but the result was substantially worse than results reported here, which are obtained without this option.

	C4.5	RIPPER	PEBLS	Rainbow	kNN	WAKNN
Syn-1	100.0	100.0	100.0	100.0	77.3	100.0
Syn-2	67.5	69.5	62.0	50.0	66.0	68.8
Syn-3	63.8	73.5	93.2	96.5	94.3	92.3
Syn-4	74.8	81.3	84.5	80.9	80.6	86.1

Table 1: Classification accuracies of different classifiers on synthetic data sets



(a) Synthetic Data Set 1 (Syn-1)

(b) Synthetic Data Set 2 (Syn-2)

(c) Synthetic Data Set 3 (Syn-3)

(d) Synthetic Data Set 4 (Syn-4)

Figure 1: Synthetic Data Sets.

Synthetic Data Sets We illustrate different cases when particular classifiers perform well and other classifiers perform bad. We have also constructed synthetic data sets according to these cases and show that our claims are true with the classification results of different classifiers. Table 1 shows the classification accuracies of different classifiers on these data sets. For the first three synthetic data sets, we have 2 classes with 200 samples each as a training set. The last synthetic data set is constructed by combining the three data sets, and thus have 6 classes with 200 samples each as a training set. Test sets of these data sets are constructed similarly.

In document data sets, some words are characteristics of a class. Consider the data set (Syn-1) shown in Figure 1 (a). There are two classes in the data set, and each class has 10 words that describe the class. Presence of any of these 20 words determine the class. In addition to these 20 words that distinguish classes, there are 80 noise words that occur randomly.

As shown in Table 1, C4.5, Rainbow, RIPPER, and PEBLS work perfectly in this data sets, as they take into consideration the class labels of the data. However, kNN performs poorly as the similarity between documents are blurred due to the noise words. In comparing two documents, the match of words from those discriminating words is considered to be the same as the match of the words from the noise words. On the other hand, WAKNN (with mutual information weight as a starting point) can overcome this problem, as the initial weight vector will have high values for those 20 discriminating words.

In documents, there exist cases when single words do not have discriminating power whereas, pair (or group) of words have discriminating power. For example, word “Clinton” could appear in many different news articles, but yet as a single word will not have much distinguishing power. When it is coupled with say “Washington-DC”, as a pair, they could become a distinguishing words for national politics article. Consider the following data set (Syn-2) illustrated in Figure 1 (b). In this data set, for each class there are 10 distinct pairs of words that determine the class. Each word occurs 40% of the all documents. However, any pair of words occur together only in one class but not in

	Source	# train	# test	# class	# words used
west-1	West Group	500	1500	10	977
west-2	West Group	300	900	10	1078
west-3	West Group	488	245	10	1035
west-4	West Group	559	280	10	887
west-5	West Group	621	311	10	1156
west-6	West Group	732	367	10	789
west-7	West Group	885	433	10	779
fbis	TREC-5	2463	1232	17	2000
trec6	TREC-5	1173	587	14	2000
reuters	Reuters-21578	6552	2581	59	2000

Table 2: Summary of data sets used.

the other class.

In this particular data set, Naive-Bayesian classifiers like Rainbow perform poorly, as the conditional probability of particular word given a class is the same for all the words. For example, given a test document with “Clinton” and “Washington-DC”, Naive-Bayesian classifiers cannot put it into the national politics category, as individual words “Clinton” and “Washington-DC” equally appear in many different categories. The failure is due to the independent assumption Naive-Bayesian classifiers make. The other classifiers would tend to perform better than Naive-Bayesian classifiers in this data set.

Another characteristics of document data sets is such that class is determined by set of words that occur more frequently in one class than another classes. Consider the following data set (Syn-3) illustrated in Figure 1 (c). In this data set, there are 50 words that occur more frequently in class A, and another 50 words that occur more frequently in class B. Furthermore, there are 100 noise words.

Decision tree based schemes like C4.5 or rule generating schemes like RIPPER do not work very well in this scenario due to the overfitting. The overfitting occurs as the number of samples is relatively small with respect to the distinguishing words. In the ideal case of decision tree (or classification rules), all the 100 distinguishing words participate in the path of decision tree (or classification rules). However, this requires training samples of size in the exponential of the distinguishing words. This is not a realistic size of training set you can expect to have.

In real situation, documents will have all these flavors of Syn-1, Syn2, and Syn3. It will not be possible to separate these types of documents from the original data sets. Consider the following data set illustrated in Figure 1 (d) where all of the above documents are concatenated. For each data set, new words introduced by other document sets are filled with zeros.

As shown in Table 1, this data set, WAKNN outperforms all other approaches. This is expected, as WAKNN performs equally well in all of the three synthetic data sets discussed. This result is confirmed with the real data sets as well.

Real Data Sets In all of the data sets, we have used stop words and stemming using Porter’s suffix-stripping algorithm [Por80]. The summary of these documents available in Table 2.

First 7 data sets are from the statutory collections of the legal document publishing division of West Group described in [CT97]. Out of 149,655 collections of documents, we selected 7 subsets of documents that have single label. For each of these data sets, we further filtered out words that occur in less than 3 documents. We then randomly selected training sets and test sets. Some of the examples of class labels of these documents include “counties”, “sales”, “worker’s compensation”, and “insurance”.

Data set $fbis$ is from the Foreign Broadcast Information Service data of TREC-5 [TRE99a]. The class labels were generated from the relevance judgment provided by TREC-5 routing query relevance ”qrels.1-243” [TRE99b]. We collected documents that have relevance judgment and selected documents that have single relevance judgment. Data set $trec$6 is from the Foreign Broadcast Information Service and LA Times data of TREC-5. The class labels are generated similarly to fbis data set using TREC-6 ad hoc query relevance ”qrels.trec6.adhoc” [TRE99b]. We randomly selected training sets and test sets for these 2 data sets. After filtering using stopping words and stemming, we further filtered the words using mutual information. We selected top 2000 words according to the mutual information of the training set.

Data set $reuters$ is from Reuters-21578 text categorization test collection Distribution 1.0 [Lew99]. We split the documents according to the modified Lewis split and selected documents with single label. After filtering words using

	C4.5	RIPPER	PEBLS	VSM	Rainbow	kNN	kNN-mut	WAKNN
west-1	85.50	84.47	78.50	86.27	84.40	76.73	86.80	**89.60**
west-2	71.30	68.33	67.80	75.22	72.11	68.33	75.89	**80.44**
west-3	79.60	75.92	72.70	82.04	80.00	70.61	80.00	**88.16**
west-4	81.80	77.14	78.60	85.00	**88.57**	73.93	81.79	85.00
west-5	84.60	89.71	86.80	89.71	85.21	84.57	90.03	**95.18**
west-6	83.70	83.38	79.80	86.92	85.29	73.57	80.38	**88.92**
west-7	80.10	80.14	71.80	81.04	81.26	74.94	81.94	**84.42**
fbis	57.10	73.94	69.80	76.14	76.38	78.49	76.54	**81.09**
trec-6	67.50	80.58	84.30	87.56	92.16	91.99	88.42	**92.67**
reuters	84.50	85.59	84.60	running	**91.04**	90.62	89.66	90.04

Table 3: Classification accuracies of different classifiers. Note that the highest accuracy for each data set is highlighted with bold font.

	against best	against kNN-mutual	against VSM	against Rainbow
west-1	+ **2.80**	+ **2.38**	+ **2.80**	+ **4.23**
west-2	+ **2.67**	+ **2.34**	+ **2.67**	+ **4.15**
west-3	+ 1.90	+ **2.47**	+ 1.90	+ **2.47**
west-4	- 1.25	+ 1.02	0.00	- 1.25
west-5	+ **2.58**	+ **2.45**	+ **2.58**	+ **4.18**
west-6	+ 0.83	+ **3.21**	+ 0.83	+ 1.47
west-7	+ 1.23	+ 0.98	+ 1.32	+ 1.23
fbis	+ 1.61	+ **2.76**	+ **3.00**	+ **2.86**
trec6	+ 0.33	+ **2.49**	+ **2.93**	+ 0.33
reuters	- 1.23	+ 0.45	running	- 1.23

Table 4: z value of WAKNN against other classifiers. Positive numbers correspond to the fact that WAKNN did better than other classifiers. Positive number greater than 1.96 shows that WAKNN is statistically better than the other classifiers and negative number less than -1.96 shows that WAKNN is statistically worse than the other classifiers. These numbers are highlighted with bold fonts.

stop words and stemming, we further selected top 2000 words according to the mutual information of the training set.

Table 3 shows the comparison of different classifiers on 10 data sets. WAKNN has the best result in 8 out of these 10 data sets. We performed a simple statistics test to see if these results are statistically significant. We performed statistical test described in [SC89], and used in [DHB95, Die98]. In this test for comparing classifier A and B, we measure

$$z = \frac{p_A - p_B}{\sqrt{2p(1-p)/n}}$$

where p_A is the error rate of classifier A on a test set, p_B is the error rate of classifier B, $p = (p_A + p_B)/2$, and n is the number of samples in the test set. We can reject the null hypothesis that two classifiers are not different in terms of performance if $|z| > Z_{0.975} = 1.96$. We calculated these statistics of WAKNN against the best results of other classifiers except kNN-mutual for each data set. Table 4 shows the results. WAKNN was the best in 8 data sets and was significantly better for 3 data sets (west-1, west-2, and west-5). For those 2 data sets in which WAKNN was not the best, it was not significantly worse than the best results. Compared against kNN-mutual, WAKNN was always better and was significantly better in 7 data sets. This shows that the weight adjustment did improve the results from the initial weight set using mutual information. VSM and Rainbow were the best among other classifiers for these data sets. When compared against VSM, WAKNN was better in all of the data sets and significantly better in 6 data sets. When compared against Rainbow, WAKNN was better in 8 data sets and significantly better in 5 data sets.

In traditional Information Retrieval systems, given multiple classes in the document set, binary classification for individual class is considered. The effectiveness of retrieval is measured in terms of recall, precision, and F-measure for each class label separately [RL94]. In these experiments (not reported here), we have confirmed that WAKNN outperforms other classifiers in terms of the micro-averaged F-measure [RL94] as well.

4 Conclusions and Directions of Future Research

In this paper, we presented a k-nearest neighbor classification algorithm that learns importance of attributes and utilizes them in the similarity measure. As our experimental results have shown, our algorithm is very effective in the text categorization task. However, a number of key issues remain to be addressed. One issue is how to avoid the local minima in the search for the best weight vector. In addition to the main objective function used in WAKNN, we might need secondary objective functions to move out of the local minima. Another possible solution to the local minima problem might be changing weights of multiple words at a time. A big challenge for this solution is how to find the set of words for the weight change. Another issue is whether the enhanced weight adjustment leads to overfitting. Even though the relatively large number of neighbors and high majority percentage in the objective function tends to reduce the risk of overfitting, this problem can be significant in many data sets. Finally, the computational cost of weight adjustment step of WAKNN is $O(cn^2)$ where c is the number of iterations in the weight adjustment step and n is the number of data points. We will investigate optimization schemes to improve upon this computational complexity.

Acknowledgment

We would like to thank Dr. Paul Thompson and Dr. Isabell Moulinier of West Group for allowing us to have the data for experiments.

References

[BGG+99] D. Boley, M. Gini, R. Gross, E.H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Partitioning-based clustering for web document categorization. *Decision Support Systems (accepted for publication)*, 1999.

[CH98] W.W. Cohen and H. Hirsh. Joins that generalize: Text classification using WHIRL. In *Proc. of the Fourth Int'l Conference on Knowledge Discovery and Data Mining*, 1998.

[Coh95] W.W. Cohen. Fast effective rule induction. In *Proc. of the Twelfth International Conference on Machine Learning*, 1995.

[CS93] S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10(1):57–78, 1993.

[CT91] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 1991.

[CT97] T. Curran and P. Thompson. Automatic categorization of statute documents. In *Proc. of the 8th ASIS SIG/CR Classification Research Workshop*, Tucson, Arizona, 1997.

[DHB95] T.G. Dietterich, H. Hild, and G. Bakiri. A comparison of ID3 and backpropagation for english text-to-speech mapping. *Machine Learning*, 18:51–80, 1995.

[Die98] T.G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7), 1998.

[Joa97] T. Joachims. A probabilistic analysis of the rocchio algorithm with TFIDF for text categorization. In *Proc. of the Fourteenth International Conference on Machine Learning*, 1997.

[Lew98] D. Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *Tenth European Conference on Machine Learning*, 1998.

[Lew99] D. D. Lewis. Reuters-21578 text categorization test collection distribution 1.0. *http://www.research.att.com/ lewis*, 1999.

[LG94] D. Lewis and W. Gale. A sequential algorithm for training text classifiers. In *SIGIR-94*, 1994.

[Low95] D.G. Lowe. Similarity metric learning for a variable-kernel classifier. *Neural Computation*, pages 72–85, January 1995.

[LR94] D. Lewis and M. Ringuette. Comparison of two learning algorithms for text categorization. In *Proc. of the Third Annual Symposium on Document Analysis and Information Retrieval*, 1994.

[McC96] Andrew Kachites McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.cmu.edu/ mccallum/bow, 1996.

[MN98] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[Por80] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Qui93] J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[RL94] E. Riloff and W. Lehnert. Information extraction as a basis for high-precision text classification. *ACM Transactions on Information Systems*, 12(3), 1994.

[Sal89] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[SC89] G.W. Snedecor and W.G. Cochran. *Statistical Methods*. Iowa State University Press, 1989.

[TRE99a] TREC. Text REtrieval conference. *http://trec.nist.gov*, 1999.

[TRE99b] TREC. Text REtrieval conference relevance judgements. *ftp://ftp-nlpir.nist.gov/pub/trec/qrels*, 1999.

[Yah99] Yahoo! Yahoo! http://www.yahoo.com, 1999.

[Yan94] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *SIGIR-94*, 1994.

[YC94] Y. Yang and C.G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems*, 12(3), 1994.

[YP97] Y. Yang and J. Pederson. A comparative study on feature selection in text categorization. In *Proc. of the Fourteenth International Conference on Machine Learning*, 1997.