

CS 545 Assignment 8

A Survey on Approaches to Improve k NN Classification Performance

Manaf H. Gharaibeh

December 16, 2009

Contents

1	Introduction	1
2	Text Categorization and k NN Overview	2
2.1	An Overview of Text Categorization ...	2
2.2	Importance of Text Categorization ...	3
2.3	k Nearest Neighbor (k NN) ...	3
3	Data Representation	4
3.1	Vector Model ...	4
3.2	Terms Weighting ...	4
3.3	Similarity Measuring ...	5
4	Improving k NN Performance using Different Term Weighting Schemes	5
4.1	Boosting k NN Supervised Term Weighting Schemes ...	5
4.2	An Improved <i>tf-idf</i> Approach ...	9
5	Distance Metric Learning for Large Margin Nearest Neighbor	10
5.1	Motivation for using Mahalanobis Distance Metric ...	10
5.2	Mahalanobis Distance ...	10
5.3	Results ...	12
6	Conclusions	13

1 Introduction

Huge amount of electronic textual information is increasingly available through the internet and organizations, making the process of retrieving data and information very difficult with out good indexing and summarizing of documents contents. Text or document categorization is one solution for the problem. Many statistical learning methods have been applied in the field of text categorization in recent years [10] [11] [12], this includes regression models, nearest neighbor classifiers, Bayes belief networks, decision trees, rule learning algorithms, neural networks, and inductive learning techniques. In this report we discuss one very important text classification algorithm which is the k nearest neighbor (k NN).

Text categorization or classification¹ (TC) refers to the process of classifying free text, to predefined categories, giving it one or more category label, in machine learning approaches to text categorization this is done by learning the system how to classify through examples. Automatic text categorization can significantly reduce the cost of manually categorization.

This report presents an overview of text categorization and the k NN algorithm in particular, and a survey of different approaches to improve the performance of k NN algorithm. Some studies suggest using different distance measuring schemes, other studies suggest different terms weighting schemes, or use different samples representation, or using a feature selection method. I will discuss two of those approaches in this survey: terms weighting, and distance metrics. My motivation for this survey is to see how researcher use different approaches to improve the popular k NN classifier. After choosing the topic of this report, I searched for recent articles about text classification with emphasis on the k NN algorithm, I tried to find articles that suggest improvements to k NN. For each chosen article, I have examined the researcher motivation, the problem he/she is trying to solve, and his/her approach to solve it, and finally the reported results. I tried to summarize and discuss all of these for each article, in addition I have incorporated additional clarifications and information from other articles or websites with related material.

The rest of this survey is organized as follows: section 2 presents an overview of text categorization, its importance, and a brief description of the k NN algorithm. Section 3 discusses the presentation of data samples in space. Section 4 presents two terms weighting approaches to improve k NN classification. Section 5 presents a different distance metric approach to improve k NN performance. And finally section 6 presents conclusions and remarks.

2 Text Categorization and k NN Overview

The following subsections describe Text categorization and its importance. Also a simple illustration for the k NN algorithm is presented.

2.1 An Overview of Text Categorization

Text categorization (TC) is a Natural Language processing (NLP) problem; it can be defined as the assignment of unclassified document to one or more predefined categories based on their content. Automatic TC is very useful in terms of time and expense that it saves, many methods and algorithms have been applied to the problem of text categorization; these methods vary in their accuracy and computation efficiency [13].

Text categorization goes through a number of stages; figure 1 shows a general overview for the framework of a typical categorization system. The pre-processing phase prepares the document for the classification process. This is similar to many natural language processing problems where tags and

¹ *Classification and categorization are synonyms and are used interchangeably in this document.*

stop words are removed. Stemming is applied to the remaining text in the document this helps reducing the feature space of the problem. The indexing step uses a weighing scheme to weight terms in the text, the next step in categorization is to select suitable feature space among the terms in the documents, this is a crucial step; the accuracy of the system depends highly on the keywords selected to represent documents, also the computation complexity depends on the number of keywords selected. Choosing a small subset of a category relative keywords or choosing uninformative keywords can lead to a deficiency in the accuracy of classification, while a very large number of keywords can make a classification algorithm inefficient in terms of time.

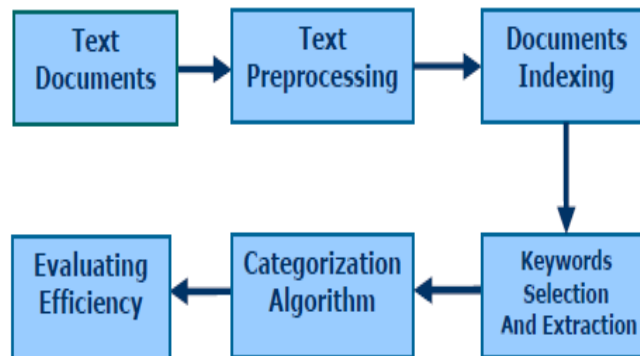


Figure 1: Overview of the categorization process

In the next step of classification, the documents are split into training and testing documents, the training documents are used to train the system to recognize different patterns of categories. Testing documents, are used to evaluate the system. The process of categorization depends on the algorithm used.

2.2 Importance of Text Categorization

There are great amounts of textual information on the internet and in computerized systems of any big organization or association, it is very expensive to manually manipulate this huge amount of information because it is time consuming and needs expensive expertise who can not be available for 24 hours a day, automatic TC helps in reducing the time needed to classify hundreds or thousands of daily arrived documents, with out the need for experts.

Text classification is an old research area, it gained more interest in the 1990's because of the increasing amount of textual information and the improvement of computer technologies which enabled a more sophisticated algorithms to be implemented, text categorization application date back to the 1960's [14], it was part of other fields like Information Retrieval (IR) and Information Filtering (IF), but became a separate research area in the 1980's.

2.3 k Nearest Neighbor (kNN)

Key nearest neighbor (*k*NN) is a statistical learning algorithm which has been studied as a pattern recognition approach for over than four decades [15]. *k*NN is known to be one of the top performing methods, many studies applied *k*NN on the benchmark Reuters corpus² [10, 11, 14, 16], these studies suggest that *k*NN with Support Vector Machines (*SVM*) outperform other methods like Linear least square fit (LLSF), Naïve Bayes (*NB*) and Neural networks (*NNet*) [10, 14, 16, 17].

² The Reuters 21578 test collection is publicly available at: <http://www.research.att.com/~lewis/reuters21578.html>

k NN is a simple, yet very efficient example based approach for text categorization, the idea of k NN can be explained as follows: given a test document to be classified, the algorithm searches for the k nearest neighbors among the pre-classified training documents based on some similarity measure, and ranks those k neighbors based on their similarity scores, the categories of the k nearest neighbors are used to predict the category of the test document by using the ranked scores of each as the weight of the candidate categories, if more than one neighbor belong to the same category then the sum of their scores is used as the weight of that category, the category with the highest score is assigned to the test document provided that it exceeds a predefined threshold, more than one category can be assigned to the test document.

One draw back in k NN is the difficulty to determine the value of k [17], a series of experiments with different k values should be conducted to determine the best value of k , another disadvantage of k NN is the complexity of computation time needed to traverse all the training documents.

3 Data Representation

Text documents should be represented in some way that enables the classifier to interpret them, an indexing method is needed to transform text documents represented by strings of characters to another interpretable representation of the documents contents, the same indexing method should be applied for both training and testing documents and for the validation set if exists. The most popular approach for data representation is the vector space model proposed by Salton and Buckley [18].

3.1 Vector Model

The vector model represents documents as vectors in the space, each document can be represented by a vector that represents the weights of the terms within the document with respect to the dimension of the space, the number of dimensions equals the number of terms or keywords used, we can represent this as a two way matrix where the columns represent terms and rows represent documents in the set, the entries of the matrix are the weights of term i in document j . In the basic two dimension Cartesian plane, a vector is represented by two points, each consist of the ordered pair x and y , to represent a vector of N terms we need N dimensions.

3.2 Terms Weighting

Although there are different weighting approaches for text indexing, they all share the following two observations:

- The more the number of times a term occurs in documents that belong to some category, the more it is relative to that category.
- The more the term appears in different documents representing different categories, the less the term is useful for discriminating between documents as belonging to different categories.

The most commonly used weighting approach is the $tf \times idf$, and the Normalized $tf \times idf$ which overcomes the problem of variant documents lengths and is represented by the following formula:

$$w_{ij} = tfidf(t_i, d_j) = \frac{f_{ij}}{\sqrt{\sum_{k=1}^M f_{kj}^2}} \times \log\left(\frac{N}{n_i}\right)$$

where N is the number of documents in the data set, M is the number of terms used in the feature space,

f_{ij} is the frequency of a term i in document j , and n_i denotes the number of documents that term i occurs in at least once. Using this formula, long documents will have a higher length normalization value and thus reducing the $tf \times idf$ value for the terms in that document, this makes them comparable with those in documents of shorter lengths.

3.3 Similarity Measuring

The most commonly used similarity measure between documents represented as vectors in r -coordinate space where r is the number of terms in the feature space is the *cosine* similarity measure. Assuming A and B are vectors representing documents j and k respectively then we can calculate the similarity between A and B using the following formula which takes into consideration that A and B may have different lengths:

$$sim(A, B) = \frac{\sum_{i=1}^r w_{ij} \times w_{ik}}{\sqrt{\sum_{i=1}^r w_{ij}^2} \times \sqrt{\sum_{i=1}^r w_{ik}^2}}$$

4 Improving k NN Performance using Different Term Weighting Schemes

In this section we present two proposed weighting schemes to improve k NN text classification performance, the first one is by using supervised weighting scheme instead of unsupervised ones, and the second approach is by using an improved version of the popular *tf-idf* weighting scheme called TFIDFIE.

4.1 Boosting k NN Supervised Term Weighting Schemes

The paper by Batal and Hauskrecht[1] discusses the use of supervised weights learning instead of conventional unsupervised weights. According to [2, 3] the most popular weighting function is the *tf-idf* which was originally introduced in the information retrieval field IR and then adopted in much of the work in the text classification work. *tf-idf* does not care about the category of the terms, i.e. it does not take in consideration how much a term is discriminant for a given category, it mainly focuses on the frequency of the term within text. The motivation of Batal and Hauskrecht in their study comes from the idea that k NN accuracy is sensitive to the distance metric used, and that giving all the terms in the feature space the same weights may bias the distance metric since this allows irrelevant, noisy, or redundant terms the same power within the distance calculations as those terms which should have a more significant effect. In particular they study the use of information gain (IG) and chi-square statistics (X^2), which are supervised schemes that consider terms weights according to their categories relation, so the terms that are more discriminant for some given category basically take higher weights. This is not a new approach in text classification however, it has been in use by many researchers for a quite long time like in [4]. The interesting issue in this paper however is the claim that k NN can outperform the support vector machine state of the art classification algorithm which most studies give it the advantage over the k NN classifier, this can be achieved by using supervised weights. In their empirical study for the effect of category skew on six feature selection methods on 36 data sets, Simeon and Hilderman found that the the highest F-measure values were obtained by bi-normal separation and information gain IG , and the highest Precision values were obtained by categorical proportional difference and chi-squared (X^2) [5].

Terms Weighting

the study included five different term weighting schemes, the first three are unsupervised weights, and the later two are supervised.

■ Unsupervised weights:

- binary weights, which in its basic form gives the weight 1 if the term exists in a document and 0 otherwise.
- Term frequency (tf), which counts the number of occurrences of a term within a document.
- Term frequency-inverse document frequency ($tf-idf$), which also counts the frequency of the term but favors terms that appear less throughout the set of documents in a data set.

■ Supervised weights:

The Chi-square and IG methods are typically used to select features in the stage of document indexing, however they are used here to weight terms. So instead of eliminating irrelevant terms the study here gives them low weights, and it gives greater weights for the more predictive terms. A brief explanation for each follows:

- Chi-square X^2 , which typically measures how an observation is different from an initial hypothesis, so in text classification if we assumed that the initial hypothesis is that the term t_k and the the class c_i are independently distributed, then X^2 measures how much the two lack independence. If the the observed probabilities are denoted by P and the expected probabilities under the independence assumption are denoted by E then X^2 is defined as in the following formula:

$$\chi^2(t_k, c_i) = \sum_{c \in \{c_i, \bar{c}_i\}} \sum_{t \in \{t_k, \bar{t}_k\}} \frac{(P(t, c) - E(t, c))^2}{E(t, c)}$$

- Information gain (IG), evaluates how much a term t_k gives information about a given class c_i , IG is calculated using the following formula:

$$IG(t_k, c_i) = \sum_{c \in \{c_i, \bar{c}_i\}} \sum_{t \in \{t_k, \bar{t}_k\}} P(t, c) \cdot \log_2 \frac{P(t, c)}{P(t) \cdot P(c)}$$

Before going to the experimental results, there are two notes about the terms weighting discussion here. This study suggests that the $tf-idf$ favors the rare terms and ignores the relationship between the terms and categories, which they consider inappropriate for text classification, however the expression “rare term” is a little bit ambiguous here, because $tf-idf$ favors the terms that have high frequency in some documents but not all documents, so it gives less weights to the terms that appear in a larger number of documents, because it consider them less discriminative, which makes sense, and even more this somehow relates the term to a given category, so the $tf-idf$ is not really blind to categories. For example the term “surgery” is expected to appear frequently in the medical articles but not as much in the sport or politics articles, so the $tf-idf$ gives it high weight, however a term like “write” can be found in many articles belonging to different categories, so it is not discriminative to any category, $tf-idf$ gives such terms low weights. And the fact that $tf-idf$ is used in most of the text classification research reflects that it is good method in practice.

The second note is that X^2 and IG both uses the class labels in their calculations here, which is convenient for the training phase because all the training documents are pre-categorized, but for a new sample, we don't know its class in advance, the study here uses probabilities relative to how much a term t_k in a new sample is discriminative for a class c_i , which means biasing kNN to that class c_i , this

kind of biasing needs more investigation.

Experiments and Results

The Study uses a text classification benchmark from CMU’s 20-Newsgroups data-set, and it tests the classification performance on three data sets, however the number of categories is very small in each of the sets, it is only 4 in the first one, 6 in the second set, and the last one has only two categories, so the results here are a little bit optimistic I think, because it is easier to classify when you have a small number of classes, the more classes you have the more overlapping the terms will be, and hence the less discriminant they become, so it will be better if a data set with a greater number of categories is used.

The documents in the data sets are preprocessed, stop words are removed, and a very light stemming is applied that removes -s, -ed, and -ing suffixes. The similarity distance measure used here is the cosine similarity. The performance measure used is the F1 score[7], which can be interpreted as a weighted average of the precision and recall, calculated globally in this study for all the categories using micro-average or macro-average. Figure 2 shows the results of k NN classification using the five different weighting schemes applied to 3 different data set, the figure shows that applying supervised weighting schemes IG and X2 gives better results than using than unsupervised ones. Also the figure shows that applying feature selection methods to the feature space to select the most relevant features improves the results also, this was also indicated by Yang and Pedersen [6] where they suggested the that the Chi-square X^2 is one of the best methods for feature selection in text classification.

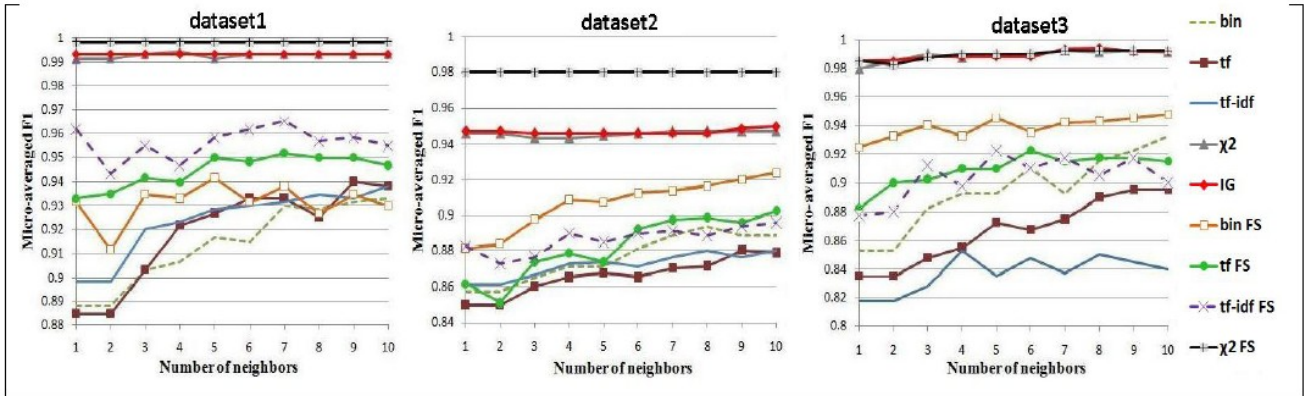


Figure 2: F1 results of applying distance weighted k NN to 3 different data sets

The study gives a justification for the better results when using Chi-square X^2 and IG for weighting by by studying the data space representation using two more measures: the intra-category similarity and the inter-category similarity. The first one measures the similarity between documents of the same category and is defined as follows:

$$\frac{1}{K} \sum_{k=1}^K \left[\frac{1}{n_k^2} \sum_{d \in c_k} \sum_{d' \in c_k} \text{COS}(d, d') \right]$$

where K indicates the number of classes, n_k indicates the number of documents in class c_k , d and d' are documents which belong to the same category, and the COS function is used to measure the similarity between them. The second measure the similarity between the documents from different categories and is defined as follows:

$$\frac{1}{\frac{K(K-1)}{2}} \sum_{k_1=1}^K \sum_{k_2=k_1+1}^K \left[\frac{1}{n_{k_1}} \frac{1}{n_{k_2}} \sum_{d \in c_{k_1}} \sum_{d' \in c_{k_2}} \text{COS}(d, d') \right]$$

The higher the intra-category and the smaller the inter-category means that it is easier to separate the categories, figure 3 shows the results of calculating those two measures for the five different weighting schemes used in the experiments. The figure shows that χ^2 and IG make the samples of one category more similar, and more distinct from other categories samples.

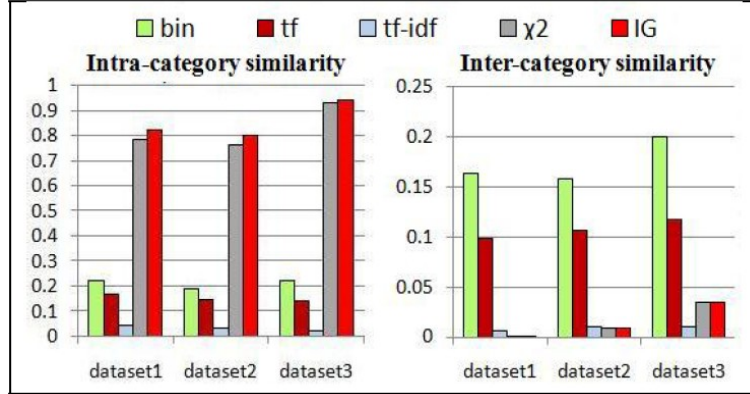


Figure 3: The intra-category similarity(left), and inter-category similarity(right)

The study in this paper also compares the performance of the k NN classifier against the Support Vector Machine SVM suggested by Vapnic in [8]. k NN is implemented with *tf-idf* and *IG* weighting methods. Figure 4 shows the comparisons results.

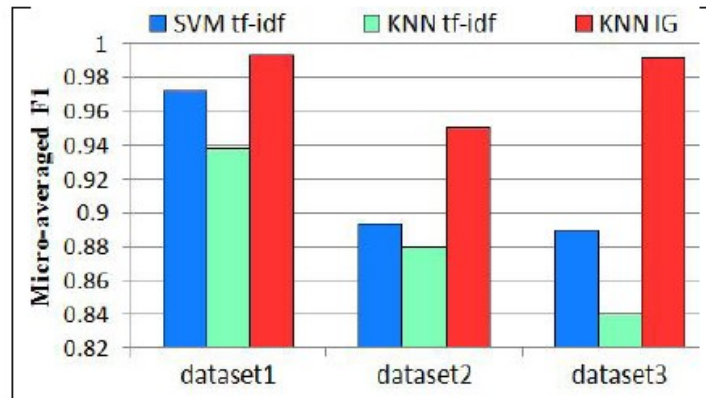


Figure 4: A comparison between SVM and k NN implemented with *tf-idf* and k NN implemented with *IG*

The results in figure 4 show that *SVM* is better than k NN implemented with *tf-idf*, however when k NN is implemented with *IG* weighting scheme it outperforms the *SVM* classifier. This test was implemented without applying any feature selection. However and again the data sets used are small and the number of categories is probably not enough to conclude that k NN with *IG* or with *chi-square* will in general outperform the *SVM* classifier. Overall the study suggests that the supervised weights improve the k NN performance because they result in high impact of the more important discriminative

terms on the distance calculations, and also because they help clustering documents of one category in the representation space.

4.2 An Improved *tf-idf* Approach

Zhang, et al. [9] proposed an approach similar to the *tf-idf* for term weighting, they called it *tfidfie*, the added “ie” at the end is for Inverted Entropy. Their motivation for this is the same as for Batal and Hauskrecht[1] in the previous discussion, where they believe that *tf-idf* does not take in consideration the distribution of the documents containing the terms to be weighted. To illustrate their idea an example of two terms t_1 and t_2 is given. t_1 and t_2 are in the same document and have the same frequency and the same document frequency, however t_1 only appears in a category while t_2 appears uniformly in all the categories. Obviously the term t_1 is more discriminative and should be given a higher weight, which does not happen with *tf-idf*, which gives the same weight for both t_1 and t_2 . For this they propose the following improved *tf-idf* to emphasize on the importance of a term to a category. The new formula for the *tf-idf* uses the information entropy H [20] defined as

$$H(t_k, d_j) = - \sum_{l=1}^{|c|} \frac{DF_{kl}}{\#_{T_r}(t_k)} \log \frac{DF_{kl}}{\#_{T_r}(t_k)}$$

where $\#_{T_r}$ denotes the document frequency of the term t_k , and DF_{kl} denotes the number of documents in category c_l in which t_k occurs, and $|c|$ is the number of categories in T_r (the training set). Based on the definition of H and the maximum entropy theorem, the more the documents that contains the term t_k are uniformly distributed in T_r the higher the value of H , on the other hand the less they are uniformly distributed the lower the value of H gets. So the original formula of *tf-idf* is modified to contain H to become term frequency, inverted document frequency, inverted entropy *tfidfie* and is defined as

$$tfidfie(t_k, d_j) = \frac{\#(t_k, d_j) \cdot \log \frac{|T_r|}{\#_{T_r}(t_k)}}{- \sum_{l=1}^{|c|} \frac{DF_{kl}}{\#_{T_r}(t_k)} \log \frac{DF_{kl}}{\#_{T_r}(t_k)}}$$

The added idea here is: the more the term exists in uniformly distributed documents among the data set the less discriminative it is and hence the less weight it should have. This idea considers the distribution of the documents that contain the term and it sounds nice, but I think that the most discriminative terms by default will not appear in uniformly distributed documents in T_r , and so the original *tf-idf* will reward those by giving them higher weights, however this might give less weight for the less discriminative terms that does not appear in many documents but are somewhat uniformly distributed among many categories documents. So overall I don't think the gains would be considerable, and the results in table 1 which contains the results of the this experiment support this claim, using the improved *tfidfie* did improve the result but by a margin of less than 2.5% which is sill an improvement.

The experiment included 4 classifiers: k NN, SVM, Naïve Bayes (NB), and Clonal Selection Algorithm Based on Antibody Density (CSABAD), which is is an improved immune algorithm proposed by the authors in [9]. The data set used is OHSCAL, which is a subset from the OHSUMED medical data set [21], the OHSCAL subset contains 11,162 documents from 10 categories: Antibodies, Carcinoma, DNA, In-Vitro, Molecular-Sequence-Data, Pregnancy, Prognosis, Receptors, Risk-Factors

and Tomography. Finally the performance measure used is the micro-average F_1 score.

average micro-averaging F_1	kNN	SVM	NB	CSABAD
<i>tfidf</i>	66.39%	71.03%	68.61%	71.46%
<i>tfidfie</i>	68.28%	73.42%	70.95%	73.17%

Table 1: Classification performance using different classification methods

5 Distance Metric Learning for Large Margin Nearest Neighbor

This paper by Weinberger and Saul [22], is the most interesting one in this survey, it has a lot of variations, and a considerable amount of information, and with 38 pages it is really a lengthy paper. The main focus in this research is to learn the Mahalanobis distance metric from labeled examples to the k NN classifier.

5.1 Motivation for using Mahalanobis Distance Metric

The authors motivation comes from the fact that k NN performance is very sensitive to the distance metrics used, and that (the typically used) Euclidean distance metric ignores any statistical regularities that can be estimated from a large number of pre-categorized set of training examples. Their purpose of using Mahalanobis is to choose target k nearest neighbors so that they belong to the same class, while separating neighbors from others classes by a large margin. The proposed classification method is called large margin nearest neighbor (LMNN). By increasing the number of neighbors that share the same label, which is done by learning a linear transformation of the input space before using the Euclidean distance metric in the k NN classifier, k NN will classify the example correctly. This linear transformation is achieved basically by minimizing the loss function throughout:

- Penalizing the large distances between the examples that belong to the same class in which they are desired to be the k nearest neighbors.
- Penalizing the the small distances between examples that belong to different classes.

So the Euclidean distance metric will be used in the linearly transformed space as if the Mahalanobis is used in the original space, which turns the problem of distance metric learning into a convex optimization problem, more about convex optimization can be found in [24]. An explanation for the Mahalanobis distance follows.

5.2 Mahalanobis Distance

Mahalanobis metric was originally introduced by P. C. Mahalanobis in 1936.[23] in the statistics field. Unlike the Euclidean distance metric, this metric considers the correlations between variables in the data set, and is useful to estimate similarities between known samples of a set and unknown samples, which is typical for the problem of text classification.

For the problem of classification, given a number of classes and a sample to be classified, the covariance matrix for each class is first calculated and then the Mahalanobis distance between the sample and each class is calculated, the sample is then assigned the label of the class with the least Mahalanobis distance.

In this paper the Mahalanobis distance is computed as the squared distance between two samples x_i and x_j represented as vectors using the following formula:

$$\mathcal{D}_{\mathbf{M}}(\vec{x}_i, \vec{x}_j) = (\vec{x}_i - \vec{x}_j)^\top \mathbf{M}(\vec{x}_i - \vec{x}_j)$$

which is similar to the Euclidean metric but now with matrix \mathbf{M} added to the distance computation. $\mathbf{M} \geq 0$ and is a positive semidefinite matrix which means it has no negative eigenvalues, with the goal that \mathbf{M} minimizes the k NN error rate. \mathbf{M} should work such that the distance between the sample to be classified with samples of the same class is less than the distance with any sample from other classes by a large margin, the following is an explanation for the matrix \mathbf{M} .

If we perform a linear transformation over the vector space X such that

$$\vec{x}' = \mathbf{L}\vec{x}$$

then the squared distance metric can be computed as:

$$D_{\mathbf{L}}(\vec{x}_i, \vec{x}_j) = \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|_2^2$$

where D_L indicates distance after linear transformation, and the linear transformation is parametrized by the matrix \mathbf{L} . The squared distance in this equation can be expressed in terms of the square matrix \mathbf{M} :

$$\mathbf{M} = \mathbf{L}^\top \mathbf{L}$$

where \mathbf{M} calculated this way is guaranteed to be a positive semidefinite matrix. The distance metric can be parametrized by the matrix \mathbf{M} or the matrix \mathbf{L} , the later eliminates the constrain that \mathbf{M} should be ≥ 0 and is a positive semidefinite matrix.

For the purpose of finding a linear transformation for the vector space in the problem of k NN classification such that the k nearest neighbors share the same label, the authors review the use of different Eigenvectors methods which are the principal components analysis (PCA), the linear discriminant analysis (LDA), and the relevant components analysis (RCA). More explanation for each of those is presented in[22].

To summarize, the model used here is built with the goal that each training input example \vec{x}_i should share the same label y_i with its k nearest neighbors, and the neighbors with different labels should be separated by a large margin. A linear transformation is learned such that it satisfies these goals by penalizing the large distances with neighbors from the same class, and penalizing small distances with neighbors with different class labels, meaning that we are looking for target neighbors that we want them to be the closest for each input sample. The term imposter is used in this paper to denote a sample l with different class label and is closer to a given sample i than a sample j with the same label as i by a margin less than or equal to one unit. This is indicated by

$$\|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|_2^2 \leq \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|_2^2 + 1$$

Figure 5 shows how LMNN linearly transform the space such that imposters are separated by some margin outside the region that established by target neighbors. As previously stated LMNN penalizes small distances with neighbors with different class labels. The figure show how LMNN makes it easier for the classifier to find the correct neighbors and hence the correct label.

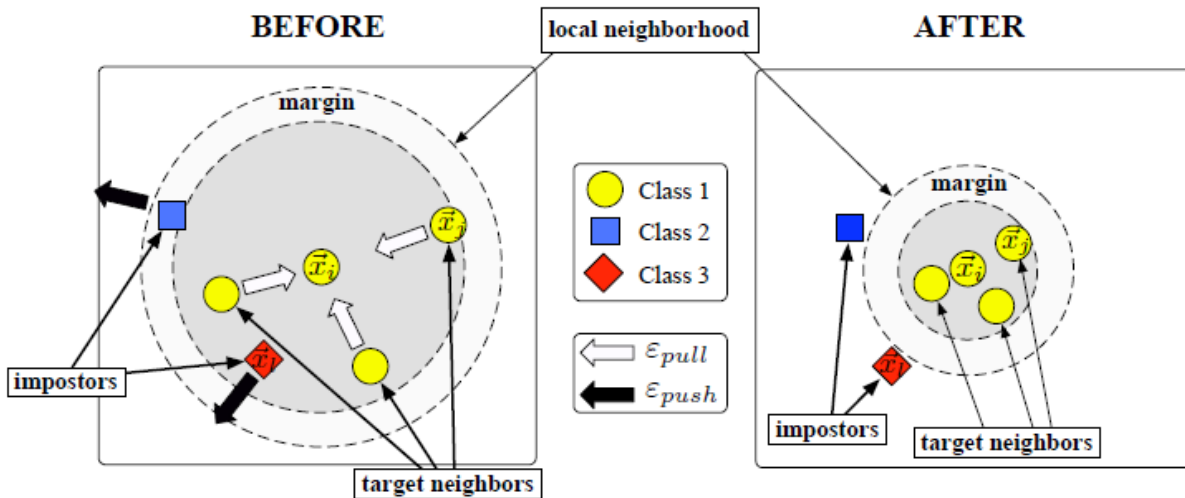


Figure 5: Sample i neighborhood before and after using LMNN

5.3 Results

LMNN classification was tested on nine data sets including data sets derived from images, speech and text, meaning they have high dimensionality, PCA was used to reduce the feature space dimensionality before training with LMNN. Figure 6 shows the main results for the five largest data sets. The upper part of the figure shows the training error rates, and the lower one shows the test error rates for the same data sets. Multiple metric LMNN (mm-lmnn), multiple passes LMNN (mp-lmnn), and lmnn(energy) are extensions designed to improve the LMNN classifier but are not cover in this survey. The experiments are applied to data sets after pre-processing with PCA to reduce dimensionality and noise, in all the cases except for the Isolet and MNIST data sets, the samples were split 70% to 30% training and testing partitions, the number of desired target neighbors k was set to 3. The authors claims that the results of LMNN with Mahalanobis distances outperforms the results of k NN implemented with Euclidean distance only. Also in most experiments pre-processing using PCA yielded better results compared to results using LDA. Also the authors claims but without a formal analysis that LMNN yields better improvements on larger data sets. Finally, as our main concern in this survey is text classification, we take a closer look at the results of classifying the 20-newsgroups data set, where the study claims an improvement over the k NN with Euclidean distance and k NN with PCA, LMNN recorded a 14.98% error rate, compared to 48.57% and 18.22% for the two k NN versions respectively. However LMNN was outperformed by the multiclass SVM which recorded 8% test error rate.

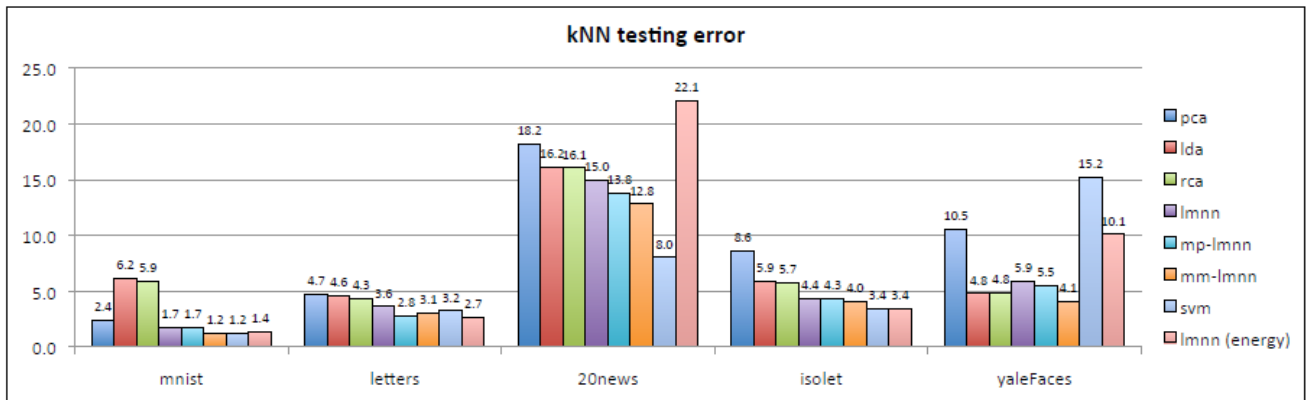
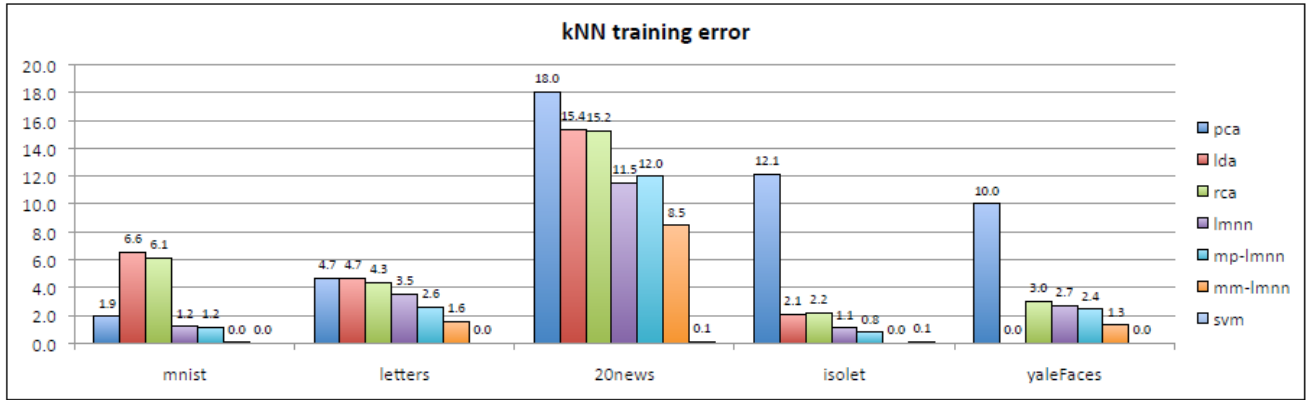


Figure 6: Results of experiments on the five large sets with different kNN implementations with and without LMNN and SVM implementation

There are still a lot of interesting details in this paper that worth to be discussed, unfortunately time does not permit that.

6 Conclusions

This survey investigated two approaches for improving k NN performance, the first one is by using different weighting schemes, and the second one is by using different distance metric. Two examples are discussed for the first approach; the first example suggests that using supervised weighting schemes like the Information Gain IG and the chi-square X^2 which are originally used as feature selection methods, improves the result of k NN classifier that only implements the $tf-idf$, were $tf-idf$ is claimed to be insensitive for how terms are related to different categories. The second terms weighting approach suggest using an improved version of $tf-idf$ called TFIDFIE, this approach uses inverted entropy to make the $tf-idf$ sensitive to the distribution of the documents that contains a given term, and hence, the more these documents are uniformly distributed among categories the less the term is important and the less weight it is given, on the other hand the less the term exists in documents of different categories the more it is discriminative and hence the higher weight it is given. For the second approach, this survey discussed the use Mahalanobis distance metric to help the k NN classifier find the correct labeled neighbors in the training phase. The approach is called large margin nearest neighbor LMNN. Given an

input sample, this approach mainly increases the distance margin with neighbors who have labels not equal to that of the input training sample by penalizing the distance between them, it also penalizes the large distances with neighbors who have the same label as the input sample. To conclude, Text classification algorithms like k NN can still be improved by carefully studying the existing research and trying to find improvements to any parts of the classification process.

References

- [1] I. Batal and M. Hauskrecht. Boosting KNN Text Classification Accuracy by using Supervised Term Weighting Schemes. CIKM'09, November 2–6, 2009, Hong Kong, China. Copyright 2009 ACM 978-1-60558-512-3/09/11.
- [2] D. Mladenić, J. Brank, M. Grobelnik, and N. Milic-Frayling. Feature Selection using Linear Classifier Weights: Interaction with Classification Models. In Proc. of ACM SIGIR, 2004.
- [3] Y. Yang and X. Liu. A re-examination of Text Categorization Methods. In Proc. SIGIR, 1999.
- [4] Y. Yang and J. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In Proc. of ICML, 1997.
- [5] M. Simeon and R. Hilderman. Feature Selection for Text Categorization. In Proc. of the 22nd Canadian Conference on Artificial Intelligence: Advances in Artificial Intelligence, Kelowna, Canada. Pages 449-252 Volum 5549. 2009.
- [6] Y. Yang and J. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In Proc. of ICML, 1997.
- [7] F. Sebastiani and C. Ricerche. Machine Learning in Automated Text Categorization. ACM Computing Surveys, 2002.
- [8] V. Vapnik. Statistical Learning Theory. Wiley-Interscience, 1998.
- [9] Qirui Zhang, Jinghua Tan, Huaying Zhou, Weiye Tao, Kejing He, "Machine Learning Methods for Medical Text Categorization," paccs, pp.494-497, 2009 Pacific-Asia Conference on Circuits, Communications and Systems, 2009
- [10] Y. Yang and X. Liu. "A Re-Examination of Text Categorization Methods". In proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99), 1999, PP. 42 - 49.
- [11] Y. Yang and J. O. Pedersen. "A Comparative Study on Feature Selection in Text Categorization". In Proceedings of ICMC-97, 14th International Conference on Machine Learning (Nashville, IN, 1997), PP. 412 - 420.
- [12] F. Sebastiani. "Machine Learning in Automated Text Categorization". ACM Computing Surveys (CSUR), 2002, 34(1): PP. 1 – 47.
- [13] A. Rafael, J. Lee. Managing Content with Automatic Document Classification. 2003.
- [14] H. Varatharajan, S. J. Maddikayala. Machine Learning For Gene Expression Analysis. (a web site) <http://dcm.cl.uh.edu/caps4gp1>.
- [15] A Tang. Text Categorization Using Support Vector Machine. An MSc thesis. 2001.
- [16] T. Joachims. "Text Categorization with Support Vector Machines: Learning with Many Relevant Features".

Proceedings of ECML-98, 10th European Conference on Machine Learning, 1998, PP. 137 – 142.

[17] J. He and A. Tan and C. Tan. "Comparative Study on Chinese Text Categorization Methods". On the PRICAI 2000 Workshop on Text and Web Mining, (Melbourne, 2000), PP. 25 - 31.

[18] G. Salton and C. Buckley. "Term Weighting Approaches in Automatic Text Retrieval". Information Processing and Management, VOL 24, 1988, PP. 513 – 523.

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

[20] MEDLINE database information. <http://healthlinks.washington.edu/howto/medline.html>, 2008.

[21] OHSUMED dataset, <ftp://medir.ohsu.edu/pub/ohsumed> , 2008.

[22] Weinberger, Kilian Q. Saul, Lawrence K., 'Distance Metric Learning for Large Margin Nearest Neighbor Classification', *J. Mach. Learn. Res.* , vol. 10, 207-244 (2009).

[23] Mahalanobis distance, Wikipedia: http://en.wikipedia.org/wiki/Mahalanobis_distance, Retrieved Dec. 2009.

[24] Convex optimization, Wikipedia: http://en.wikipedia.org/wiki/Convex_optimization, Retrieved Dec. 2009.