Boosting KNN Text Classification Accuracy by using Supervised Term Weighting Schemes

Iyad Batal
Department of Computer Science
University of Pittsburgh
iyad@cs.pitt.edu

Milos Hauskrecht
Department of Computer Science
University of Pittsburgh
milos@cs.pitt.edu

ABSTRACT

The increasing availability of digital documents in the last decade has prompted the development of machine learning techniques to automatically classify and organize text documents. The majority of text classification systems rely on the vector space model, which represents the documents as vectors in the term space. Each vector component is assigned a weight that reflects the importance of the term in the document. Typically, these weights are assigned using an information retrieval (IR) approach, such as the famous *tf-idf* function. In this work, we study two weighting schemes based on information gain and chi-square statistics. These schemes take advantage of the category label information to weight the terms according to their distributions across the different categories. We show that using these supervised weights instead of conventional unsupervised weights can greatly improve the performance of the k-nearest neighbor (KNN) classifier. Experimental evaluations, carried out on multiple text classification tasks, demonstrate the benefits of this approach in creating accurate text classifiers.

Categories and Subject Descriptors

I.7 [**DOCUMENT AND TEXT PROCESSING**]: General

General Terms

Algorithms, Experimentation, Performance

Keywords

Text Classification, Supervised Weights, K-Nearest Neighbors

1. INTRODUCTION

With the rapid growth of online information, text classification has become one of the key techniques for handling and organizing text data. Many interesting applications are based on text classification such as: junk mail detection, articles filtering based on long-term standing interests (e.g., filtering all CIKM articles that talk about text classification), document organization (e.g. organizing the medical journals of Medline), hierarchical categorization of web pages (such as Yahoo!'s topic hierarchy), etc...

CIKM'09, November 2–6, 2009, Hong Kong, China.

The construction of a text classifier usually involves two phases:

1. **Indexing phase**: This phase converts each document into a vector in the term space using the popular vector space model (VSM) [10]. If words are chosen to be the terms, the dimensionality of each vector is the number of all distinct words in the corpus (known as the *vocabulary size*). Each vector component is assigned a *weight* related to the estimated importance of the corresponding term in the document. In this phase, term selection/extraction techniques can be optionally applied to reduce the dimensionality of the space.
2. **Learning phase**: This phase uses the training data to build a classification model capable of deciding the categories of future unlabeled documents.

A large number of statistical machine learning techniques have been proposed to improve the learning phase of the text classifier [1, 11, 16, 15]. Traditionally, category label information associated with the training documents is used in the learning phase to build the classification model. In addition, many researchers proposed using this information to perform supervised dimensionality reduction during indexing [8, 7, 4, 18]. However, the weights of the terms are mostly assigned in an unsupervised way. Currently, the most popular term weighting function is *tf-idf* [9], originally proposed in the information retrieval (IR) field, and subsequently adopted in most text classification research [8, 7, 11, 17, 6].

This work focuses on instance-based learning for text classification. We study ways of incorporating category information into term weights in order to improve the distance/similarity metric of the KNN classifier. KNN is an extremely simple yet surprisingly effective method for text classification [16, 17, 6]. However, the caveat of dealing with KNN is that its performance critically depends on the quality of the distance metric. Giving all terms the same weight may bias the distance metric because this allows irrelevant, redundant, or noisy terms to have as much effect on distance computation as other terms. This observation motivated us to study the effect of term weighting schemes on KNN performance.

We show that substituting conventional unsupervised weights with supervised weights greatly improves KNN performance. In particular, we study using information gain (*IG*) and chi-square statistics (χ^2), which have been previously found to be excellent for term selection [18], to weight the terms. Intuitively, these supervised schemes give high weights for terms that are discriminative among the categories.

To illustrate the idea, suppose that we have documents from two categories: "computer science" and "art". Clearly, if we want to classify a document that contains the word "algorithm", this word should give a strong indication that the document probably belongs to the "computer science" category. Thus, "algorithm" should have a high influence on KNN distance metric. Supervised weights help to automatically focus more on important terms and less on com-

mon uninformative terms. This ability does not exist in the *tf-idf* approach because it is a category-blind weight.

The contributions of this paper can be summarized as follows:

1. We study using *IG* or χ^2 scores, traditionally applied for feature selection, to weight the terms. We show that utilizing these supervised weights, instead of conventional unsupervised weights, significantly boosts KNN performance.
2. We empirically demonstrate how the supervised weights improve space representation by clustering the documents according to their categories.
3. We show that when KNN uses the supervised weights, it can outperform SVM, the state-of-the-art text classifier.

2. TERM WEIGHTING

In this study, we compare five different weighting schemes: binary weights (*bin*), term frequency (*tf*), term frequency-inverse document frequency (*tf-idf*), chi-square (χ^2) and information gain (*IG*). These weighting schemes can be divided into two main groups: supervised weights and unsupervised weights, depending on whether or not they use the category label information available in the training dataset when calculating the weight.

2.1 Traditional unsupervised weights

bin is the simplest way to score the terms of a document. A term gets a weight of 1 if it exists in the document, and 0 otherwise.

tf counts the number of occurrences of a term in a document. *tf* is usually normalized to prevent a bias towards longer documents.

tf-idf [9] originated from the IR field to score and rank the documents relevant to a user query. Later on, *tf-idf* has been adopted by the majority of text classification researchers [8, 7, 11, 17, 6]. The *idf* factor favors **rare** terms over frequent terms. In IR, the *idf* assumption is reasonable since a rare term is usually important to discriminate a document from a set of similar documents. However, in text classification, this assumption is inappropriate because it completely ignores the presence of labeled training data.

2.2 Supervised weights

We study two supervised methods based on chi-square statistics and information gain. Conventionally, these methods are used to do feature filtering during document indexing [18]. However, this paper instead studies using these scores to weight the terms of the document instead of selecting them. These supervised weights change the term space representation of the documents by assigning large weights to predictive terms and low weights to irrelevant terms, which has a great effect on KNN performance.

χ^2 is frequently used to measure how the results of an observation differ from the results expected according to an initial hypothesis. In text classification, the initial hypothesis is that term t_k and category c_i are independently distributed. Thus, the χ^2 statistics measures the lack of independence between t_k and c_i. Denoting the observed probabilities by P and the expected probabilities under the independence assumption by E, χ^2 is defined as:

$$\chi^2(t_k, c_i) = \sum_{c \in \{c_i, \bar{c_i}\}} \sum_{t \in \{t_k, \bar{t_k}\}} \frac{(P(t,c) - E(t,c))^2}{E(t,c)} \quad (1)$$

IG is an information-theoretic function which measures the amount of information that can be obtained about one random variable by observing another. In the text classification context, *IG* evaluates how much information term t_k contains about category c_i

$$IG(t_k, c_i) = \sum_{c \in \{c_i, \bar{c_i}\}} \sum_{t \in \{t_k, \bar{t_k}\}} P(t,c) \,.\, log_2 \frac{P(t,c)}{P(t).P(c)} \quad (2)$$

After computing χ^2 or *IG*, we multiply them with *tf* in order to incorporate the "local" importance of the term in its document.

We can see from equations (1 and 2) that calculating the χ^2 or *IG* weight of a term in a document requires knowing the category label of that document. However, this label is unknown for the documents we want to classify. Therefore, we define the category independent weight of a term in an unlabeled document ($w_{unlabeled}(t_k)$) to be the maximum of all individual category-related weights ($max_{c_j}\{w(t_k, c_j)\}$).

The rationale for doing this is that if a new document d' contains a term t_k that is highly discriminative for a specific category c_i, then $w_{unlabeled}(t_k) = w(t_k, c_i)$. Therefore, the occurrence of t_k helps making d' more similar to the documents of category c_i, i.e. it biases KNN towards choosing category c_i.

3. EXPERIMENTAL EVALUATION

3.1 Data

In our experiments, we draw the text classification benchmarks from the CMU's 20-Newsgroups dataset. We tested the classification performance on three different datasets:

Dataset1: This dataset consists of documents from the following 6 categories: *baseball*, *computer graphics*, *hockey*, *motorcycles*, *space* and *christianity*. We select randomly 100 documents from each of these categories.

Dataset2: This dataset consists of documents from 4 categories: *general politics*, *guns*, *Mideast* and *religion*. We select 200 documents from each category. All documents in this dataset belong to the same discussion group: *talk.**. Therefore, categories in dataset2 are more similar to each other than the categories in dataset1.

Dataset3: Dataset3 is a binary classification task to differentiate documents that talk about *PC* computers and documents that talk about *MAC* computers. The contents of these documents are very similar since they are all about computer hardware (drawn the discussion group *comp.sys.**). Each category contains 200 documents.

Pre-Processing: Before indexing the documents, we remove standard stop words. Concerning stemming, we only apply the first step of porter's stemming algorithm, which removes the -s, -ed and -ing suffixes from the words. The resulting vocabulary size for dataset1 is *16,184*, for dataset2 is *18,194* and for dataset3 is *8,496*.

3.2 Classifier

This study focuses on comparing the effectiveness of the different term weighting schemes using the k-nearest neighbor (KNN) classifier. KNN has the advantages of being a non-parametric and non-linear classifier. Experiments in [16, 17, 6] showed KNN to be one of the top-performing text classification methods.

To make KNN work, we must define a distance/similarity measure for comparing documents. We use the cosine similarity [9], which measures the similarity between two documents by finding the cosine of the angle between their vectors in the term space. The similarity ranges from 0 to 1, where 0 indicates independence (the vectors are orthogonal) and 1 indicates a perfect match.

We use the distance weighted version of KNN, which weights the vote of each neighbor by its similarity to the document.

3.3 Performance measures

The effectiveness of a text classifier is usually measured using the traditional IR measures of precision and recall. First, the precision and recall are calculated separately for each category. Then the global precision π and recall ρ are obtained by averaging individual precisions and recalls using either micro-average or macro-average. since our datasets are balanced (all categories have the same gen-

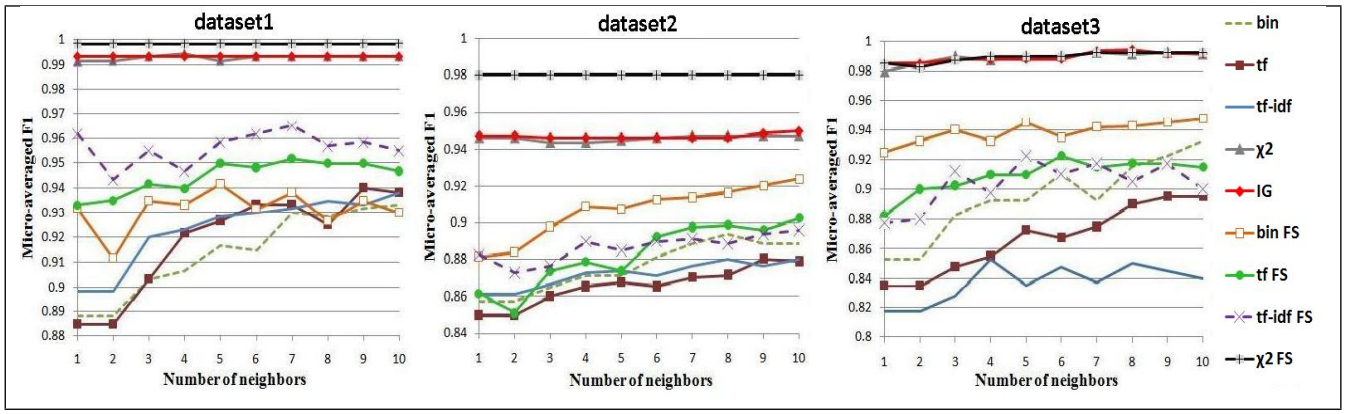

Figure 1: Comparing the F1 score of distance-weighted KNN for the different weighting schemes on dataset1, dataset2 and dataset3

erality), the results of micro-averaging and macro-averaging are almost identical [12]. In order to combine the effect of π and ρ in one measure, we use the well-known F_1 score [12].

In this work, we experimentally demonstrate how the supervised schemes tend to cluster the documents in the vector space according to their categories. In order to assess this clustering ability for the different weighting schemes, we define two measures: the *intra-category* similarity and the *inter-category* similarity.

The intra-category similarity measures the average similarity between the documents from the same category and is defined as:

$$\frac{1}{K}\sum_{k=1}^{K}\left[\frac{1}{{n_k}^2}\sum_{d\in c_k}\sum_{d'\in c_k} COS(d,d')\right] \quad (3)$$

Where K is the number of categories and n_k is the number of documents from category c_k.

The expression between brackets calculates the average pairwise cosine similarity for the documents of the same category (c_k). These quantities are added and divided by K to obtain the average intra-category similarity for all categories.

Similarly, the inter-category similarity measures the average similarity between the documents from different categories.

$$\frac{1}{\frac{K(K-1)}{2}}\sum_{k_1=1}^{K}\sum_{k_2=k_1+1}^{K}\left[\frac{1}{n_{k_1}}\frac{1}{n_{k_2}}\sum_{d\in c_{k_1}}\sum_{d'\in c_{k_2}} COS(d,d')\right] \quad (4)$$

The expression between brackets calculates the average pairwise cosine similarity between the documents of category c_{k_1} and the documents of category c_{k_2}. The total inter-category similarity is obtained by adding the similarity between every pair of categories and dividing the sum by $K(K-1)/2$.

A good space representation should have a **high** intra-category similarity and a **low** inter-category similarity to indicate that the categories can be easily separated by the classification algorithm. Thus, we define the *AR ratio* (intra-category similarity to inter-category similarity ratio)

$$AR\ ratio = \frac{intra\text{-}category\ similarity}{inter\text{-}category\ similarity} \quad (5)$$

Clearly, the bigger the *AR ratio* for a weighting scheme, the better it represents the documents in the space.

3.4 Results

In our experiments, we report the micro-average F1 function on the test set, using a 5-fold cross validation scheme.

3.4.1 KNN classification

In this section, we compare the 5 different weighting schemes: *bin*, *tf*, *tf-idf*, *IG* and χ^2. We also report their effectiveness in conjunction with feature selection (denoted as FS). We perform feature selection by scoring terms according to χ^2 and selecting the top 10% terms to represent the documents. Using *IG* for feature filtering gave almost the same performance (for all schemes) as using χ^2 for feature filtering (this agrees with the findings in [18] that *IG* and χ^2 are highly correlated). Besides, the performance of IG with FS was very similar to the performance of χ^2 with FS. Thus, we omit IG FS from the graphs to improve their readability.

Figure 1 shows the performance plots of KNN on dataset1, dataset2 and dataset3 (from left to right). The x-axis represents the number of KNN neighbors and the y-axis represents the micro-averaged F_1 measure. In all figures, we can see a big performance gap between the classifier that uses the supervised weights (*IG* or χ^2) and the classifier that uses the unsupervised weights (*bin*, *tf* or *tf-idf*) for all different numbers of neighbors. For instance, the F1 measure of 5NN (using 5 neighbors) on dataset1 is 0.928 when using *tf-idf*, while it is 0.992 when using χ^2 and 0.993 when using *IG*. On dataset3, using the supervised weights instead of *tf-idf* improves 5NN performance from 0.835 to 0.99.

The graphs also show that the accuracy improves after applying feature selection. This suggests that a lot of the terms are indeed just noisy features. For example, applying χ^2 to both weight and select the terms (χ^2 FS) achieves an F1 of 0.98 on dataset2.

We can see that the simple binary weights can sometimes outperform *tf-idf* weights, as in the case of dataset3. This shows that *tf-idf* are not optimized to score the terms for text classification tasks.

3.4.2 Space representation

In order to understand the reason behind the superior classification accuracy of the supervised weights, we examine more closely the way each of the weighting schemes maps the documents in the space. We use the two measures we defined in section 3.3: the intra-category similarity (equation 3) and the inter-category similarity (equation 4) to assess the quality of the space representation. A good weighting scheme should maximize the similarity between the documents of the same category and minimize the similarity between the documents of different categories. In these experiments, we use the category information of all documents (no test set).

From Figure 2, we can see that χ^2 and *IG* (the last two columns) always increase the intra-category similarity and decrease the inter-category similarity as opposed to *bin*, *tf* and *tf-idf* unsupervised weights (the first three columns). For instance, applying *IG* on dataset1 (the first set of columns) achieves an intra-category sim-

ilarity of 0.81 and an inter-category similarity of 0.0018, whereas applying *tf-idf* on the same dataset results in an intra-category similarity of 0.044 and an inter-category similarity of 0.007. Thus, using *IG* instead of *tf-idf* increase the *AR ratio* about 75 times!

These results support the conjecture that the supervised weights improve documents representation in the term space.

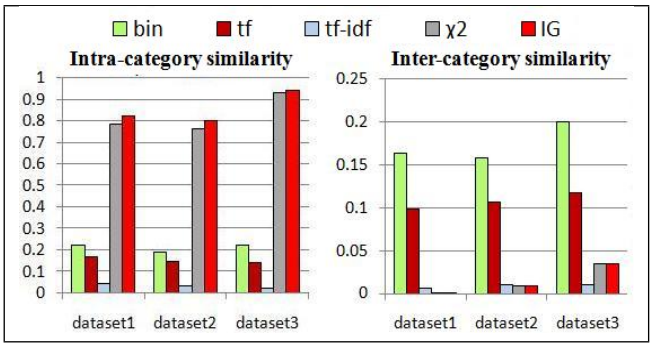

Figure 2: The intra-category similarity (left) and the inter-category similarity (right) for *bin*, *tf*, *tf-idf*, χ^2 and *IG* on the three datasets

3.4.3 Comparison with SVM

In this section, we compare the performance of KNN against SVM [13], which is considered the top performing text classification algorithm [6, 17].

SVM works by learning a linear combination of the features in order to define the decision hyperplane. Previous studies [3, 5] who applied different weighing schemes with SVM did not get much improvement because SVM itself is optimized to learn the term weights. SVM is also known to be quite robust in the presence of many features [6], and the extensive experiments in [2] confirmed this fact by observing either no improvement or small degradation in SVM performance after applying feature selection.

Consequently, we compare our approach against linear SVM with *tf-idf* weights on the full vocabulary (without applying feature selection), as defined in the original paper [6]. In order to perform multi-class classification, we adopt the *one-against-all* approach, using the continuous values of SVM decision functions to make the classification (as defined by Vapnik in [14]).

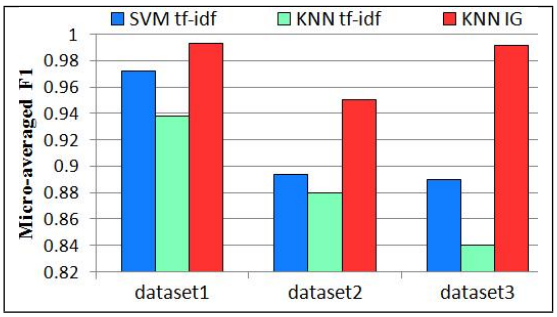

Figure 3: comparison with SVM

Figure 3 compares the performance of SVM with *tf-idf* weights, KNN with *tf-idf* weights and KNN with *IG* weights. We can see that, when we use the conventional *tf-idf* weights, SVM has higher F1 than KNN. This agrees with the findings of [6, 17], which empirically show that SVM is the top performing text classifier, followed by the KNN classifier. However, by switching to the *IG* weights, KNN is able to outperform SVM on all three datasets. For example, using KNN IG instead of SVM on dataset3 (a binary classification task) increases F1 from 0.89 to 0.9925 (almost a perfect classification). This clearly shows that appropriate supervised weights can make KNN a very effective text classifier.

4. CONCLUSIONS

This paper closely examines the aspects of document representation during the indexing phase of a text classifier, focusing on the KNN classifier. KNN is known to be a very effective text classification algorithm [16, 17, 6]. However, its most serious drawback is a modeling issue: how to define a good distance metric in order to find the "nearest" neighbors of a test document?

Many research papers have studied applying feature selection techniques in text classification to discard "useless" term [8, 4, 18]. But why not using these scores to weight the terms instead of selecting them. In this paper, we experimentally show that using supervised weights, like information gain or chi-square, can greatly boost KNN accuracy, as opposed to using conventional unsupervised weights. The reason for this superior performance is that:

1. Supervised weights give a high impact to important (discriminative) terms on the distance calculation. In comparison, *tf-idf* weights terms inappropriately by favoring rare terms.
2. Supervised weights cluster the documents in the space according to their categories.

5. REFERENCES

[1] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003.

[2] J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic. Interaction of Feature Selection Methods and Linear Classification Models. In *Proc. of the ICML Workshop on Text Learning*, 2002.

[3] F. Debole and F. Sebastiani. Supervised Term Weighting for Automated Text Categorization. In *Proc. of ACM Symposium on Applied Computing*, 2003.

[4] G. Forman. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

[5] G. Forman. BNS Feature Scaling: An Improved Representation over tf-idf for SVM Text Classification. In *Proc. of CIKM*, pages 263–270, 2008.

[6] T. Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevent Features. In *Proc. of European Conference on Machine Learning*, 1998.

[7] T. Liu, Z. Chen, B. Zhang, W. Ma, and G. Wu. Improving Text Classification using Local Latent Semantic Indexing. In *Proc. of ICDM*, 2004.

[8] 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.

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

[10] G. Salton, A. Wong, and C. Yang. A Vector Space Model for Automatic Indexing. *Commun. ACM*, 1975.

[11] R. Schapire and Y. Singer. Boostexter: a Boosting-Based System for Text Categorization. *Journal of Machine Learning Research*, 2000.

[12] F. Sebastiani and C. Ricerche. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 2002.

[13] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, NY, 1995.

[14] V. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[15] E. Wiener, J. Pedersen, and A. Weigend. A Neural Network Approach to Topic Spotting. In *Proc. of SDAIR*, 1995.

[16] Y. Yang. An evaluation of Statistical Approaches to Text Categorization. *Journal of Information Retrieval*, 1999.

[17] Y. Yang and X. Liu. A re-examination of Text Categorization Methods. In *Proc. SIGIR*, 1999.

[18] Y. Yang and J. Pedersen. A Comparative Study on Feature Selection in Text Categorization. In *Proc. of ICML*, 1997.