A re-examination of text categorization methods

Yiming Yang and Xin Liu
School of Computer Science, Carnegie Mellon University
Pittsburgh, PA 15213-3702, USA; www.cs.cmu.edu/~yiming/

Abstract

This paper reports a controlled study with statistical significance tests on five text categorization methods: the Support Vector Machines (SVM), a k-Nearest Neighbor (kNN) classifier, a neural network (NNet) approach, the Linear Least-squares Fit (LLSF) mapping and a Naive Bayes (NB) classifier. We focus on the robustness of these methods in dealing with a skewed category distribution, and their performance as function of the training-set category frequency. Our results show that SVM, kNN and LLSF significantly outperform NNet and NB when the number of positive training instances per category are small (less than ten), and that all the methods perform comparably when the categories are sufficiently common (over 300 instances).

1 Introduction

Automated text categorization (TC) is a supervised learning task, defined as assigning category labels (pre-defined) to new documents based on the likelihood suggested by a training set of labelled documents. It has raised open challenges for statistical learning methods, requiring empirical examination of their effectiveness in solving real-world problems which are often high-dimensional, and have a skewed category distribution over labelled documents. Topic spotting for newswire stories, for example, is one the most commonly investigated application domains in the TC literature. An increasing number of learning approaches have been applied, including regression models[9, 32], nearest neighbor classification[17, 29, 33, 31, 14], Bayesian probabilistic approaches [25, 16, 20, 13, 12, 18, 3], decision trees[9, 16, 20, 2, 12], inductive rule learning[1, 5, 6, 21], neural networks[28, 22], on-line learning[6, 15] and Support Vector Machines [12].

While the rich literature provides valuable information about individual methods, clear conclusions about cross-method comparison have been difficult because often the published results are not directly comparable. For example, one cannot tell whether the performance of NNet by Wiener et al.[28] is statistically significantly better or worse than the performance of SVM by Joachims[12] because different data collections were used in the evaluations of those methods, and no statistical significance analysis was conducted to verify the impact of the difference in data on the performance variation of these classifiers. Naive Bayes classifiers have exhibited relatively poor performance in previous studies [16, 20, 12]; on the other hand, some recent papers have claimed that NB methods "perform surprisingly well" and are "gaining popularity lately"[13, 18, 3]. It is difficult to understand the claimed strengths of those Naive Bayes methods because the evaluations either used non-comparable performance measures, or were tested only on selected subsets of benchmark collections (e.g., the top ten most common categories over the total of ninety). It is not even clear what has been claimed precisely; does "surprisingly well" mean statistically significant better than other methods, or not statistically different from the top-performing classifiers ever published, or worse than the best ones but still better than the expectation of the authors? If the claims were specified, then what empirical evidence has been found so far? These questions have not been addressed well.

Another open question for TC research is how robust methods are in solving problems with a skewed category distribution. Since categories typically have an extremely non-uniform distribution in practice[30], it would be meaningful to compare the performance of different classifiers with respect to category frequencies, and to measure how much the effectiveness of each method depends on the amount of data available for training. Evaluation scores of specific categories have been often reported[28, 5, 15, 13, 12]; however, performance analysis as a function of the rareness of categories has been seldom seen in the TC literature. Most commonly, methods are compared using a single score, such as the accuracy, error rate, or averaged F_1 measure(see Section 2 for definition) over all category assignments to documents. A single-valued performance measure can be either dominated by the classifier's performance on common categories or rare categories, depending on how the average performance is computed (e.g., "micro-averaging" versus "macro-averaging"). Nevertheless, no matter how the performance average is computed, a single score prohibits fine-grained analysis with respect the training-set frequencies of categories.

In this paper we address the above evaluation problems by conducting a controlled study on five well-known text categorization methods: NNet, SVM, NB, kNN and LLSF. All of these methods were published with relatively strong performance scores in previous evaluations and a partial comparison of some of them has been made[12, 31], but they

have not been directly compared together in a controlled study with thorough statistical significance analysis, which is the focus of this paper. Specifically, this paper contains the following new contributions:

- Provides directly comparable results of the five methods on the new benchmark corpus, Reuters-21578. Currently, published results of LLSF, NNet and NB on this corpus (with the full set of categories) are not available. For kNN, published results[12, 14] are available but are "mysteriously" lower than the results by others on a previous version of this collection[31]. As for SVM, the published results do not contain sufficient details for statistical significance analysis.

- Proposes a variety of statistical significance tests for different standard performance measures (including F_1 measures for category assignments), and suggests a way to jointly use these tests for cross-method comparison.

- Observes the performance of each classifier as a function of the training-set category frequency, and analyzes the robustness of classifiers in dealing with a skewed category distribution.

2 Task, Corpus and Performance Measures

To make our evaluation results comparable to most of the published results in TC evaluations, we chose topic spotting of newswire stories as the task and the Reuters-21578 corpus for the data. This corpus has become a new benchmark lately in TC evaluations, and is the refined version of several older versions, namely Reuters-22173 and Reuters-21450, on which many TC methods were evaluated[10, 16, 1, 28, 6, 33, 22, 31], but the results on the older versions may not be directly comparable to the results on the new version. For this paper we use the ApteMod version of Reuters-21578, which was obtained by eliminating unlabelled documents and selecting the categories which have at least one document in the training set and the test set. This process resulted in 90 categories in both the training and test sets. After eliminating documents which do not belong to any of these 90 categories, we obtained a training set of 7769 documents, a test set of 3019 documents, and a vocabulary 24240 unique words after stemming and stop word removal. The number of categories per document is 1.3 on average. The category distribution is skewed; the most common category has a training-set frequency of 2877, but 82% of the categories have less than 100 instances, and 33% of the categories have less than 10 instances. Figure 1 shows the category distribution in this training set.

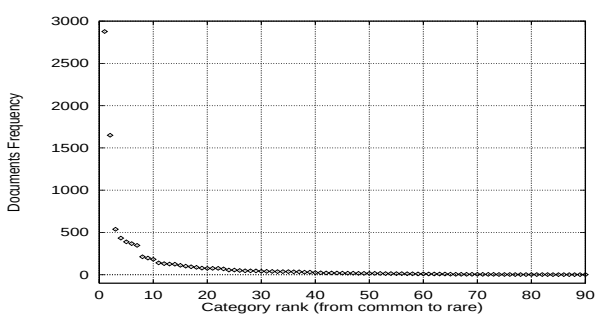

Figure 1: Category distribution in Reuters-21578 AptoMod

For evaluating the effectiveness of category assignments by classifiers to documents, we use the standard recall, precision and F_1 measure. Recall is defined to be the ratio of correct assignments by the system divided by the total number of correct assignments. Precision is the ratio of correct assignments by the system divided by the total number of the system's assignments. The F_1 measure, initially introduced by van Rijsbergen[26], combines recall (r) and precision (p) with an equal weight in the following form:

$$F_1(r,p) = \frac{2rp}{r+p}.$$

These scores can be computed for the binary decisions on each individual category first and then be averaged over categories. Or, they can be computed globally over all the $n \times m$ binary decisions where n is the number of total test documents, and m is the number of categories in consideration. The former way is called *macro-averaging* and the latter way is called *micro-averaging*. The micro-averaged F_1 have been widely used in cross-method comparisons while macro-averaged F_1 was used in some cases[15]. It is understood that the micro-averaged scores (recall, precision and F_1) tend to be dominated by the classifier's performance on common categories, and that the macro-averaged scores are more influenced by the performance on rare categories. Providing both kinds of scores is more informative than providing either alone, as we show in our evaluation and cross method comparison (Section 5).

We also use error as an additional measure, which is defined to be the ratio of wrong assignments by the system divided by the total number of the system's assignments ($n \times m$).

3 Classifiers

3.1 SVM

Support Vector Machines (SVM) is a relatively new learning approach introduced by Vapnik in 1995 for solving two-class pattern recognition problems[27]. It is based on the Structural Risk Minimization principle for which error-bound analysis has been theoretically motivated[27, 7]. The method is defined over a vector space where the problem is to find a decision surface that "best" separates the data points in two classes. In order to define the "best" separation, we need to introduce the "margin" between two classes. Figures 2 and 3 illustrate the idea. For simplicity, we only show a case in a two-dimensional space with linearly separable data points, but the idea can be generalized to a high dimensional space and to data points that are not linearly separable. A decision surface in a linearly separable space is a hyperplane. The solid lines in figures 2 and 3 show two possible decision surfaces, each of which correctly separates the two groups of data. The dashed lines parallel to the solid ones show how much one can move the decision surface without causing misclassification of the data. The distance between each set of those parallel lines are referred to as "the margin". The SVM problem is to find the decision surface that maximizes the margin between the data points in a training set.

More precisely, the decision surface by SVM for linearly separable space is a hyperplane which can be written as

$$\vec{w} \cdot \vec{x} - b = 0$$

$\vec{x}$ is an arbitrary data point (to be classified), and the vector $\vec{w}$ and the constant b are learned from a training set of

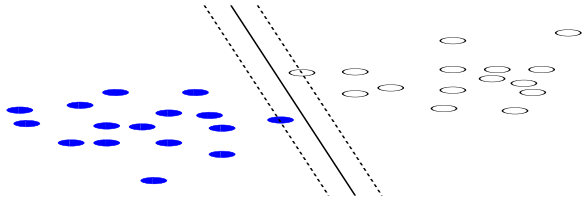

Figure 2: A decision line (solid) with a smaller margin which is the distance between the two parallel dashed lines.

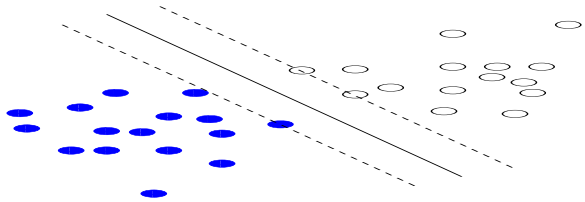

Figure 3: The decision line with the maximal margin. The data points on the dashed lines are the Support Vectors.

linearly separable data. Letting $D = \{(y_i, \vec{x_i})\}$ denote the training set, and $y_i \in \{\pm 1\}$ be the classification for $\vec{x}$ (+1 for being a positive example and -1 for being a negative example of the given class), the SVM problem is to find $\vec{w}$ and b that satisfies the following constraints

$$\vec{w} \cdot \vec{x_i} - b \geq +1 \quad for \quad y_i = +1 \tag{1}$$

$$\vec{w} \cdot \vec{x_i} - b \leq -1 \quad for \quad y_i = -1 \tag{2}$$

and that the vector 2-norm of $\vec{w}$ is minimized.

The SVM problem can be solved using quadratic programming techniques[27, 7, 23]. The algorithms for solving linearly separable cases can be extended for solving linearly non-separable cases by either introducing soft margin hyperplanes, or by mapping the original data vectors to a higher dimensional space where the new features contains interaction terms of the original features, and the data points in the new space become linearly separable[27, 7, 23]. Relatively efficient implementations of SVM include the SVM^{light} system by Joachims[12] and the Sequential Minimal Optimization (SMO) algorithm by Platt[24].

An interesting property of SVM is that the decision surface is determined only by the data points which have exactly the distance $\frac{1}{\|\vec{w}\|}$ from the decision plane. Those points are called the support vectors, which are the only effective elements in the training set; if all other points were removed, the algorithm will learn the same decision function. This property makes SVM theoretically unique and different from many other methods, such as kNN, LLSF, NNet and NB where all the data points in the training set are used to optimize the decision function. It would be interesting to know whether or not this theoretical distinction leads to significant performance differences between SVM and other methods in practice.

Joachims recently applied SVM to text categorization, and compared its performance with other classification methods using the Reuters-21578 corpus. His results show that SVM outperformed all the other methods tested in his experiments and the results published by that time[1]. While this comparison is quite informative, several points are missing or questionable:

- Thorough statistical significance tests were lacking.
- Performance analysis with respective to category distribution, especially on rare categories, was not provided.
- The kNN result reported by him is lower than the kNN results by others[31].

For addressing the above points, we decided to re-test SVM using the SVM^{light} system by Joachims[2] and our own version of kNN.

3.2 kNN

kNN stands for k-nearest neighbor classification, a well-known statistical approach which has been intensively studied in pattern recognition for over four decades[8]. kNN has been applied to text categorization since the early stages of the research [17, 29, 11]. It is one of the the top-performing methods on the benchmark Reuters corpus (the 21450 version, Apte set); the other top-performing methods include LLSF by Yang, decision trees with boosting by Apte et al., and neural networks by Wiener et al. [2, 12, 14, 31, 28].

The kNN algorithm is quite simple: given a test document, the system finds the k nearest neighbors among the training documents, and uses the categories of the k neighbors to weight the category candidates. The similarity score of each neighbor document to the test document is used as the weight of the categories of the neighbor document. If several of the k nearest neighbors share a category, then the per-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. By sorting the scores of candidate categories, a ranked list is obtained for the test document. By thresholding on these scores, binary category assignments are obtained. The decision rule in kNN can be written as:

$$y(\vec{x}, c_j) = \sum_{\vec{d_i} \in kNN} sim(\vec{x}, \vec{d_i}) y(\vec{d_i}, c_j) - b_j$$

where $y(\vec{d_i}, c_j) \in \{0, 1\}$ is the classification for document $\vec{d_i}$ with respect to category c_j (y = 1 for YES, and y = 0 for NO); $sim(\vec{x}, \vec{d_i})$ is the similarity between the test document $\vec{x}$ and the training document $\vec{d_i}$; and b_j is the category-specific threshold for the binary decisions. For convenience, we use the cosine value of two vectors to measure the similarity between of the two documents, although other similarity measures are possible. The category-specific threshold b_j is automatically learned using a "validation set" of documents. That is, we used a subset of the training documents (not used the test documents) to learn the optimal threshold for each category. By optimal, we mean the threshold that yielded the best F_1 score on the validation documents.

Note that there is a difference between the thresholding method above and the thresholding method used by Joachims in his kNN experiment. Joachims simply sorted the confidence scores per test document and assigned the top-ranking category as the correct category. This simple method does not allow the system to assign multiple categories to any document and is not necessarily the optimal strategy for kNN, or any classifier, because documents often

[1] Apte et al. later published better results of a decision tree approach using boosting[2].

[2] The SVM^{light} system is publicly available via http://www-ai.cs.uni-dortmund.de/FORSCHUNG/VERFAHREN/SVM_LIGHT/svm_light.eng.html.

have more than one category[31]. We suspect this simplification by Joachims is the reason for the low performance of his kNN; this assertion was confirmed by our experiments with both versions of kNN on Reuters-21578 (see the results in Section 5).

3.3 LLSF

LLSF stands for Linear Least Squares Fit, a mapping approach developed by Yang[32]. A multivariate regression model is automatically learned from a training set of documents and their categories. The training data are represented in the form of input/output vector pairs where the input vector is a document in the conventional vector space model (consisting of words with weights), and output vector consists of categories (with binary weights) of the corresponding document. By solving a linear least-squares fit on the training pairs of vectors, one can obtain a matrix of word-category regression coefficients:

$$F_{LS} = \arg\min_{F} \|FA - B\|^2$$

where matrices A and B present the training data (the corresponding columns is a pair of input/output vectors), and matrix F_{LS} is the solution matrix, defining a mapping from an arbitrary document to a vector of weighted categories. By sorting these category weights, a ranked list of categories is obtained for the input document. By thresholding on these category weights, category assignments to the input document are obtained. Again, the system automatically learn the optimal threshold for each category, which has the same definition as in kNN.

Although LLSF and kNN differ statistically, we have found these two methods had similar performance in all the applications where we compared these two methods, including the categorization of Reuters news stories, MEDLINE bibliographical abstracts and Mayo Clinic patient-record diagnoses[32, 29, 31]. What we have not compared yet is their robustness in dealing with rare categories; this is the one of the main foci in this study.

3.4 NNet

Neural network (NNet) techniques have been intensively studied in Artificial Intelligence[19]. NNet approaches to text categorization were evaluated on the Reuters-21450 corpus by Wiener et al. [28] and Ng. et al. [22], respectively. Wiener et al. tried both a perceptron approach (without a hidden layer) and three-layered neural networks (with a hidden layer). Ng et al. only uses perceptrons. Both systems use a separate neural network per category, learning a non-linear mapping from input words (or more complex features such as singular vectors of a document space) to a category. Wiener's experiments suggested some advantage for combining a multiple-class NNet (for higher-level categories) and many two-class networks (for lowest-level categories), but they did not compare the performance of using a multiple-class NNet alone to using a two-class NNet for each category.

In our experiments with NNet on Reuters 21578, a practical consideration is the training cost. That is, training NNet is usually much more time consuming than the other classifiers. It would be too costly to train one NNet per category, we then decided to train one NNet on all the 90 categories of Reuters. In this sense, our NNet approach is not exactly the same as the previous reported ones (we will leave the comparison for future research). We also decided to use a hidden layer with k nodes, where k is empirically chosen (Section 5). We implemented our own NNet system for efficient handling of sparse document vectors.

3.5 NB

Naive Bayes (NB) probabilistic classifiers are commonly studied in machine learning[19]. An increasing number of evaluations of NB methods on Reuters have been published[16, 20, 13, 3, 18]. The basic idea in NB approaches is to use the joint probabilities of words and categories to estimate the probabilities of categories given a document. The naive part of NB methods is the assumption of word independence, i.e., the conditional probability of a word given a category is assumed to be independent from the conditional probabilities of other words given that category. This assumption makes the computation of the NB classifiers far more efficient than the exponential complexity of non-naive Bayes approaches because it does not use word combinations as predictors.

There are several versions of the NB classifiers. Recent studies on a *multinomial mixture model* have reported improved performance scores for this version over some other commonly used versions of NB on several data collections, including Reuters-21578[18, 3]. This improved model, however, was only evaluated on a few common categories (the 10 most common ones out of the total of 90 categories) of Reuters; its results therefore do not allow a complete comparison to those previously reported for NB methods or other methods on the full set of Reuters categories. Another confusing aspect of the recent evaluations with NB is a non-conventional "accuracy" measure – the proportion of the correct category assignments among the total of n assignments (n is the number of test documents) where each document is assigned to one and only one category[13, 3, 18]. This narrowly defined "accuracy" is indeed equivalent to the standard precision under the one-category-per-document assumption on classifiers, and also equivalent to the standard recall assuming that each document has one and only one correct category. It is not equivalent, however, to the standard definition for accuracy in text categorization literature, which is the proportion of correct assignments among the binary decisions over all category/document pairs[26, 16]. The standard accuracy measure is well-defined for documents with multiple categories; the narrowly defined "accuracy" is not. The latter leads to confusion and non-comparable performance measures in text categorization evaluations on several collections, contributing to the difficulty of cross-method and/or cross-collection comparisons.

To provide comparable results of NB on Reuters-21578, we ran the multinomial mixture model of NB by McCallum[3], and evaluated its output using the standard performance measures introduced in Section 2.

4 Significance Tests

We designed a set of significance tests for comparing two systems using various performance measures. We will define the tests first, and then discuss their suitability with respect to the performance measures.

4.1 Micro sign test (s-test)

This is a sign test designed for comparing two systems, A and B, based on their binary decisions on all the docu-

[3] The NB classifiers by McCallum et al. are publicly available in the *Bow* library at cmu via http://www.cs.cmu.edu/ mccallum/bow/.

ment/category pairs. We use the following notation:

- N is the number of binary decisions by each system, i.e., the product of the number of test documents and the number of categories;
- $a_i \in \{0, 1\}$ is the measure of success for system A on the ith decision (i = 1, 2 ... N), where 1 means correct and 0 means incorrect;
- $b_i \in \{0, 1\}$ is the measure of success for system B on the ith decision;
- n is the number of times that a_i and b_i differ;
- k is the number of times that a_i is larger than b_i.

The null hypothesis is $k = 0.5n$, or k has a binomial distribution of $Bin(n, p)$ where $p = 0.5$. The alternative hypothesis is that k has a binomial distribution of $Bin(n, p)$ where $p > .5$, meaning that system A is better than system B. If $n \leq 12$ and $k >= 0.5n$, the P-value (1-sided) is computed using the binomial distribution under the null hypothesis:

$$P(Z \geq k) = \sum_{i=k}^{n} \binom{n}{i} \times 0.5^n.$$

Symmetrically, if $n \leq 12$ and $k < 0.5n$, the P-value for the other extreme is computed using the formula

$$P(Z \leq k) = \sum_{i=0}^{k} \binom{n}{i} \times 0.5^n.$$

The P-value indicates the significance level of the observed evidence against the null hypothesis, i.e., system A is better (or worse) than system B.

If $n > 12$, the P-value (1-sided) can be approximately computed using the standard normal distribution for

$$Z = \frac{k - 0.5n}{0.5\sqrt{n}}.$$

4.2 Macro sign test (S-test)

This sign test is designed for comparing two systems, A and B, using the paired F_1 values for individual categories. We use the following notation:

- M is the number of unique categories;
- $a_i \in [0, 1]$ is the F_1 score of system A on the ith category (i = 1, 2 ... M);
- $b_i \in [0, 1]$ is the F_1 score of system B on the ith category (i = 1, 2 ... M);
- n is the number of times that a_i and b_i differ;
- k is the number of times that a_i is larger than b_i.

The test hypotheses and the P-value (1-sided) computation are the same as those as in the micro s-test.

4.3 Macro t-test (T-test)

This is a t-test for comparing two systems, A and B, using the paired F_1 values for individual categories. For this we use the same notation as defined for S-test, and add the following items:

- $d_i = a_i - b_i$ is the difference of a_i from b_i;
- $\bar{d}$ is the simple average of the d_i values for $i = 1, 2, \ldots, n$.

The null hypothesis is $\bar{d} = 0$. The alternative hypothesis is $\bar{d} > 0$. If $n \leq 40$, the P-value is computed using the t-distribution with the degree of freedom (d.f.) of $n - 1$ for

$$T \geq \frac{\bar{d}}{s.e.(\bar{d})};$$

otherwise, the standard normal distribution is used instead.

4.4 Macro t-test after rank transformation

To compare systems A and B based on the F_1 values after *rank transformation*[4], in which the F_1 values of the two systems on individual categories are pooled together and sorted, then these values are replaced by the corresponding ranks. To make a distinction from the T-test above, we refer to this test as T'-test. We use the following the notation:

- a'_i is the rank of the F_1 score of system A on the ith category (i = 1, 2 ... M);
- b'_i is the rank of the F_1 score of system B on the ith category (i = 1, 2 ... M);
- $d'_i = a'_i - b'_i$ is the rank difference of a'_i from b'_i;
- n is the number of times that $a'_i \neq b'_i$;
- $\bar{d}'$ is the simple average of the d'_i values for $i = 1, 2, \ldots, n$.

The null hypothesis is $\bar{d}' = 0$. The alternative hypothesis is $\bar{d}' > 0$. If $n \leq 40$, compute the P-value (1-sided) for

$$T >= \frac{\bar{d}'}{s.e.(\bar{d}')}$$

using the t-distribution with a degree of freedom (d.f.) of $n - 1$; otherwise, the standard normal distribution is used instead.

4.5 Comparing proportions (p-test)

For the performance measures which are proportions, such as recall, precision, error or accuracy, we compare the performance scores of systems A and B as below.

- Let p_a and p_b be the performance scores by systems A and B, respectively,
- Let n_a and n_b be the numbers of *trials* in the two samples that are used to evaluation systems A and B, respectively. The definition of n_a or n_b depends on the performance measures:
  - for recall, it is the number of true YESes for categories;
  - for precision, it is the number assigned YESes by the system; and

– for accuracy or error, it is the number of document-category pairs.

- Compute $p = \frac{n_a \times p_a + n_b \times p_b}{n_a + n_b}$, the observed proportion of the total of $n = n_a + n_b$ trials.

The null hypothesis is $p_a = p_b = p$. The alternative hypothesis is $p_a > p_b$. If $n \leq 40$, compute the P-value (1-sided) for

$$Z = \frac{p_a - p_b}{\sqrt{p(1-p)(1/n_a + 1/n_b)}}$$

using the t-distribution with a degree of freedom (d.f.) of $n - 1$; otherwise, use the standard normal distribution instead. The p-test allow systems A and B to be evaluated using two different test collections as well as using a same test collection. In the case of $n_a = n_b$, the computation is simplified as $p = \frac{p_a + p_b}{2}$, and $Z = \frac{p_a - p_b}{\sqrt{2p(1-p)/n}}$.

Among the testing methods described above, s-test and p-test are designed to evaluate the performance of systems at a *micro* level, i.e., based on the pooled decisions on individual document/category pairs. On the other hand, S-test, T-test and T'-test are designed to evaluate at a *macro* level, using the performance scores on each category as the unit measure. Among these three macro tests, S-test may be more robust for reducing the influence of outliers, but risks being insensitive (or not sufficiently sensitive) in performance comparison because it ignores the absolute differences between F_1 values. The T-test is sensitive to the absolute values, but could be overly sensitive when F_1 scores are highly unstable, e.g., those for low-frequency categories. The T'-test is a compromise between the two extremes; it is less sensitive than T-test to outliers, but more sensitive than the sign test because it reserve the order of distinct F_1 values. None of the tests is "perfect" for all the performance measures, or for performance analysis with respect to a skewed category distribution, so using them jointly instead using one test alone would be a better choice.

In related literature, sign tests were reported by Cohen for method comparison based on micro-level category assignments[5], and by Lewis et al. for a comparison based on paired F_1 values of individual categories[15].

5 Evaluation

5.1 Experiments set up

We applied statistical feature selection at a preprocessing stage for each classifier, using either a χ^2 statistic or *information gain* criterion to measure the word-category associations, and the predictiveness of words (features). Different feature-set sizes were tested, and the size that optimized the global F_1 score for a classifier was chosen for that classifier. As a result, we selected 1000 features for NNet, 2000 features for NB, 2415 features for kNN and LLSF, and 10000 features for SVM.

Other empirical settings were:

- The k in kNN was set to 45. This choice was based on our previous parameter optimization (learned from training data) on Reuters-21450[31].
- The number of singular value used for LLSF computation was set to 500, which is also based on previous parameter optimization (learned from training data) on Reuters-21450[31].
- The number of hidden units in the middle layer of NNet was set to 64. This choice produces the best F_1 score for NNet on a validation set (a part of the training data) of Reuters-21578 when we varied the number of the hidden units between 16, 64 and 160.
- For SVM we tested the linear and non-linear models offered by SVM_{light}, and obtained a slightly better result with the linear SVM than with the non-linear models. We use the linear version as the representative for SVM in the cross-method comparison.

5.2 Results

Table 1 summarizes the global performance scores. Several points in this table are worth discussion.

- The micro-averaged F_1 score (.8599) of SVM is slightly lower than the best of the SVM scores (.860-.864) reported by Joachims, possibly because of a difference in our term weighting scheme for document presentation, or due to minor differences in data preparation. Joachims used the within-document frequency of terms (tf) directly, while we used $\log(tf)$ instead. This difference is not significant.
- Our micro-averaged F_1 score (.8567) for kNN is noticeably higher than the kNN score (0.823) reported by Joachims. We also tested the simplified kNN (following Joachims) which assign only the top-ranking category to each document, and obtained a F_1 score of .8140. These contrasting tests suggest that this simplification is neither optimal nor necessary for kNN.
- Our micro-averaged F_1 score (.7956) for NB is significantly higher than the NB score (0.720) obtained by Joachims. According to the analysis by McCallum at al.[18] who implemented both models, the *multinomial mixture model* which we tested is better than the *multivariate Bernoulli model* which Joachims tested. So our experiment confirmed McCallum's conclusion. However, the better model of NB still underperforms SVM, KNN and LLSF.

5.3 Cross-classifier Comparison

Table 2 summarizes the statistical significance tests. Using each column of this table, one can obtain a complete or partial order of the classifiers. The micro-level analysis (**s-test**) on pooled binary decisions suggests that

$$SVM > kNN \gg \{LLSF, NNet\} \gg NB$$

where the classifiers with insignificant performance differences are grouped into one set. On the other hand, the macro-level analyses (**S-test, T-test** and **T'-test**) on the F_1 scores suggest a somewhat different grouping and ordering of the classifiers:

$$\{SVM, kNN, LLSF\} \gg \{NB, NNet\}$$

These inconsistent grouping and orderings of classifiers reflect the biases in the performance measures. The micro-level significance test is dominated by the performance of the classifiers on common categories, while the macro-level significance tests are more reflective of the performance of the classifiers on rare categories. This does not necessarily mean that the significance tests are invalid; on the contrary,

Table 1: Performance summary of classifiers

method	miR	miP	miF1	maF1	error
SVM	.8120	.9137	.8599	.5251	.00365
KNN	.8339	.8807	.8567	.5242	.00385
LSF	.8507	.8489	.8498	.5008	.00414
NNet	.7842	.8785	.8287	.3765	.00447
NB	.7688	.8245	.7956	.3886	.00544

miR = micro-avg recall; miP = micro-avg prec.;
miF1 = micro-avg F1; maF1 = macro-avg F1.

Table 2: Statistical significance test results

sysA	sysB	s-test	S-test	T-test	T'-test
SVM	kNN	$>$	$\sim$	$\sim$	$\sim$
SVM	LLSF	$\gg$	$\sim$	$\sim$	$\sim$
kNN	LLSF	$\gg$	$\sim$	$\sim$	$\sim$
SVM	NNet	$\gg$	$\gg$	$\gg$	$\gg$
kNN	NNet	$\gg$	$\gg$	$\gg$	$\gg$
LLSF	NNet	$\sim$	$\gg$	$\gg$	$\gg$
NB	kNN	$\ll$	$\ll$	$\ll$	$\ll$
NB	LLSF	$\ll$	$\ll$	$\ll$	$\ll$
NB	SVM	$\ll$	$\ll$	$\ll$	$\ll$
NB	NNet	$\ll$	$\sim$	$\sim$	$\sim$

"$\gg$" or "$\ll$" means P-value ≤ 0.01;
"$>$" or "$<$" means $0.01 <$ P-value ≤ 0.05;
"$\sim$" means P-value > 0.05.

Table 3: p-test applied to multiple measures

sysA	sysB	miR	miP	error
SVM	kNN	$\ll$	$\gg$	$\sim$
SVM	LLSF	$\ll$	$\gg$	$\ll$
kNN	LLSF	$<$	$\gg$	$<$
SVM	NNet	$\gg$	$\gg$	$\ll$
kNN	NNet	$\gg$	$\sim$	$\ll$
LLSF	NNet	$\gg$	$\ll$	$<$
NB	kNN	$\ll$	$\ll$	$\gg$
NB	LLSF	$\ll$	$\ll$	$\gg$
NB	SVM	$\ll$	$\ll$	$\gg$
NB	NNet	$\sim$	$\ll$	$\gg$

it means that those different tests provide complementary analyses about the classifiers. Moreover, one can combine evidence from different tests. For example, when the S-test, T-test and T'-test all agree on a P-value range (e.g., $\gg$), one can be more confident about the suggested significance than the case when they do not agree.

Table 3 shows additional micro-level performance analysis using p-test with multiple measures. Error-rate based comparison leads to an ordered list of equivalent classes

$$\{SVM, kNN\} > LLSF > NNet \gg NB$$

where "$>$" or "$\gg$" indicates a better classifier (with smaller error) on the left-hand side than the one on the right-hand side. The order among classifiers suggested by the error-based tests is similar to the one observed in the micro-level sign tests, except that SVM and kNN are grouped together here, and that LLSF and NNet are no longer grouped. P-test on recall (miR) and precision (miP) can also be informative when the observations are jointly used. For example, since NB $\ll$ kNN in both recall and precision, we conclude that kNN is significantly better; on the other hand, for kNN versus SVM, we cannot decide which one is better because the p-test outcomes on recall and precision are not in agreement.

Figures 4 and 5 compare the performance curves of the five classifiers with respect to the training-set frequency of categories. These curves are obtained by dividing the horizontal axis into equal-sized intervals, averaging the per-category F_1 scores per interval for each classifier, and interpolating the per-interval average scores. Figure 4 focuses on the training set frequency in the range from 1 to 60 (covering 67% of the total unique categories), where NNet and NB are clearly worse than the other three, but these three are less easy to rank. Figure 5 shows the performance curves on the full range of training-set frequencies of categories, where the effectiveness of all the classifiers are more similar to each other on common categories (with a training-set frequency of 300 or higher), compared to their relative performance on rare categories.

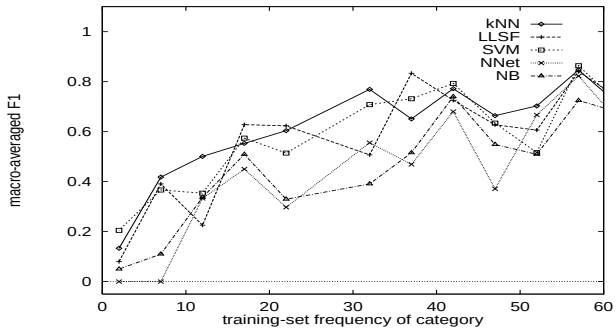

Figure 4: Performance curves on rare categories.

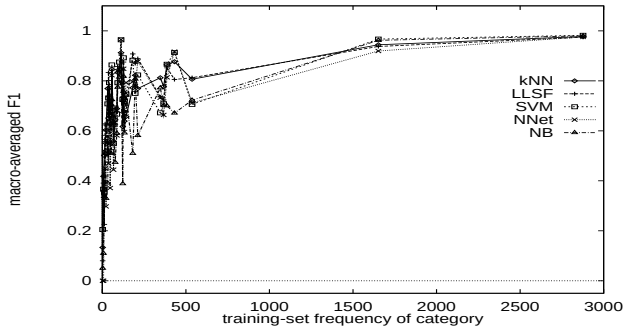

Figure 5: Performance curves on all the categories.

6 Conclusions

In this paper we presented a controlled study with significance analyses on five well-known text categorization methods. Our main conclusions are:

- Significance analyses can be applied to both a micro-level and macro-level evaluation of text categorization systems, and jointly used for cross-method comparison.
- The outcome of a significance test depends on the choice of performance measure, the sensitivity of the test, and the training-set frequency of categories being tested.
- For the micro-level performance on pooled category assignments, both a sign test and an error-based proportion test suggest that SVM and kNN significantly

outperform the other classifiers, while NB significantly underperforms all the other classifiers.

- With respect to the macro-level (category-level) performance analysis using F_1, all the significance tests we conducted suggest that SVM, kNN and LLSF belong to the same class, significantly outperforming NB and NNet.

Acknowledgment

We would like to thank Bora Cenk Gazen for verifying our results in the statistical significance tests, and Yue Pan for extending the NB method to allow category ranking and multiple labels per document.

References

[1] C. Apte, F. Damerau, and S. Weiss. Towards language independent automated learning of text categorization models. In *Proceedings of the 17th Annual ACM/SIGIR conference*, 1994.

[2] C. Apte, F. Damerau, and S. Weiss. Text mining with decision rules and decision trees. In *Proceedings of the Conference on Automated Learning and Discorery, Workshop 6: Learning from Text and the Web*, 1998.

[3] L. Douglas Baker and Andrew K. Mccallum. Distributional clustering of words for text categorization. In *Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 96–103, 1998.

[4] D. Berry and B.W. Lindgren. *Statistics: Theory and Methods*. Brooks/Cole, Pacific Grove, California, 1990.

[5] William W. Cohen. Text categorization and relational learning. In *The Twelfth International Conference on Machine Learning (ICML'95)*. Morgan Kaufmann, 1995.

[6] William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996. 307-315.

[7] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[8] Belur V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. McGraw-Hill Computer Science Series. IEEE Computer Society Press, Las Alamitos, California, 1991.

[9] N. Fuhr, S. Hartmanna, G. Lustig, M. Schwantner, and K. Tzeras. Air/x - a rule-based multistage indexing systems for large subject fields. In 606-623, editor, *Proceedings of RIAO'91*, 1991.

[10] P.J. Hayes and S. P. Weinstein. Construe/tis: a system for content-based indexing of a database of new stories. In *Second Annual Conference on Innovative Applications of Artificial Intelligence*, 1990.

[11] Makato Iwayama and Takenobu Tokunaga. Cluster-based text categorization: a comparison of category search strategies. In *Proceedings of the 18th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*, pages 273–281, 1995.

[12] Thorsten Joachims. Text Categorization with Support Vector Machines: Learning with Many Relevant Features. In *European Conference on Machine Learning (ECML)*, pages 137–142, Berlin, 1998. Springer.

[13] D. Koller and M. Sahami. Hierarchically classifying documents using very few words. In *The Fourteenth International Conference on Machine Learning (ICML'97)*, pages 170–178, 1997.

[14] W. Lam and C.Y. Ho. Using a generalized instance set for automatic text categorization. In *Proceedings of the 21th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, pages 81–89, 1998.

[15] David D. Lewis, Robert E. Schapire, James P. Callan, and Ron Papka. Training algorithms for linear text classifiers. In *SIGIR '96: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996. 298-306.

[16] D.D. Lewis and M. Ringuette. Comparison of two learning algorithms for text categorization. In *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval (SDAIR'94)*, 1994.

[17] B. Masand, G. Linoff, and D. Waltz. Classifying news stories using memory based reasoning. In *15th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'92)*, pages 59–64, 1992.

[18] 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.

[19] Tom Mitchell. *Machine Learning*. McGraw Hill, 1996.

[20] I. Moulinier. Is learning bias an issue on the text categorization problem? In *Technical report, LAFORIA-LIP6, Universite Paris VI*, 1997.

[21] I. Moulinier, G. Raskinis, and J. Ganascia. Text categorization: a symbolic approach. In *Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval*, 1996.

[22] H.T. Ng, W.B. Goh, and K.L. Low. Feature selection, perceptron learning, and a usability case study for text categorization. In *20th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*, pages 67–73, 1997.

[23] Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. In *A.I. Memo*. MIT A.I. Lab, 1996.

[24] J. Platt. Sequetial minimal optimization: A fast algorithm for training support vector machines. In *Technical Report MST-TR-98-14*. Microsoft Research, 1998.

[25] K. Tzeras and S. Hartman. Automatic indexing based on bayesian inference networks. In *Proc 16th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'93)*, pages 22–34, 1993.

[26] C.J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.

[27] V. Vapnic. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.

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

[29] Y. Yang. Expert network: Effective and efficient learning from human decisions in text categorization and retrieval. In *17th Ann Int ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'94)*, pages 13–22, 1994.

[30] Y. Yang. Sampling strategies and learning efficiency in text categorization. In *AAAI Spring Symposium on Machine Learning in Information Access*, pages 88–95, 1996.

[31] Y. Yang. An evaluation of statistical approaches to text categorization. *Journal of Information Retrieval*, 1(1/2):67–88, 1999.

[32] Y. Yang and C.G. Chute. An example-based mapping method for text categorization and retrieval. *ACM Transaction on Information Systems (TOIS)*, 12(3):252–277, 1994.

[33] Y. Yang and J.P. Pedersen. A comparative study on feature selection in text categorization. In Jr. D. H. Fisher, editor, *The Fourteenth International Conference on Machine Learning*, pages 412–420. Morgan Kaufmann, 1997.