Ambiguity Measure Feature Selection Algorithm

Saket S.R. Mengle

Information Retrieval Lab, Illinois Institute of Technology, Chicago, Illinois, U.S.A., saket@ir.iit.edu

Nazli Goharian

Information Retrieval Lab, Illinois Institute of Technology, Chicago, Illinois, U.S.A., nazli@ir.iit.edu

With the increasing number of digital documents, the ability to automatically classify those documents both efficiently and accurately is becoming more critical and difficult. One of the major problems in text classification is the high dimensionality of feature space.

With the increasing number of digital documents, the ability to automatically classify those documents both efficiently and accurately is becoming more critical and difficult. One of the major problems in text classification is the high dimensionality of feature space. We present the ambiguity measure (AM) feature-selection algorithm, which selects the most unambiguous features from the feature set. Unambiguous features are those features whose presence in a document indicate a strong degree of confidence that a document belongs to only one specific category.We apply AM feature selection on a naïve Bayes text classifier.We favorably showthe effectiveness of our approach in outperforming eight existing featureselection methods, using five benchmark datasets with a statistical significance of at least 95% confidence. The support vector machine (SVM) text classifier is shown to perform consistently better than the naïve Bayes text classifier. The drawback, however, is the time complexity in training a model. We further explore the effect of using the AM feature-selection method on an SVM text classifier. Our results indicate that the training time for the SVM algorithm can be reduced by more than 50%, while still improving the accuracy of the text classifier. We favorably show the effectiveness of our approach by demonstrating that it statistically significantly (99% confidence) outperforms eight existing feature-selection methods using four standard benchmark datasets.

Introduction

There is an overflow of unorganized digital data in today's world. Vast volumes of digital text are available via the World Wide Web (WWW), news feeds, electronic mails, corporate databases, medical patient records and digital libraries. The problem of classifying and storing these documents poses a significant challenge. Large companies filter incoming e-mail and store them in folders or route them to concerned departments. News agencies also use classification tools for filtering or routing the news from different sources to their appropriate client. Other applications of text classification are in the field of knowledge-base extraction, e-commerce and information extraction. Companies spend significant resources on classifying documents manually. The feasibility of manual classification decreases as the number of documents increases over time. As the number of documents is large, a fast and scalable automatic classifier is needed to classify the existing and incoming documents accurately and efficiently. We propose, design, develop and evaluate one such classifier.

Text classification involves scanning through the text documents, and assigning categories to documents to reflect their content (Yang, 1999). One of the major characteristics of text classification is the high dimensionality of feature set (Mladenić & Grobelnik, 1998). The feature set for a dataset consists of the unique terms in training documents. However, the number of features in the text classification dataset is prohibitively high for many learning algorithms. Hence, it is highly desirable to reduce the feature set without sacrificing categorization accuracy. Feature selection is formally defined in (Galavotti & Sebastiani, 2000) as "the activity of selecting, from the set of r distinct features (i.e., words) occurring in the collection, the subset of $r' << r$ features that are most useful for compactly representing the meaning of the documents." Feature-selection methods are used to achieve two objectives:

1. To reduce the size of the feature set to optimize the classification efficiency.
2. To reduce noise in feature set to optimize the classification effectiveness.

Most existing feature selection algorithms such as *odds ratio* (Mladenić & Grobelnik, 1998), *information gain* (Quinlan J, 1986), *Chi-Squared* (Yang & Pedersen, 1997), bi-normal separation (Forman, 2003) and *tficf* (Chih & Kulathuramaiyer, 2004) calculate a score based on the probability that a feature belongs to a given category and the probability that a feature does not belong to the other categories. These algorithms perform poorly on the unbalanced text classification datasets. The nature of unbalanced datasets is such that few categories have significantly more training documents than most of the categories, and hence, the term frequency of many features appearing in these few categories is significantly higher than their frequency in other categories. Moreover, if such terms have the same term frequency in two or more categories, the feature can not confidently point to a given category. Thus, such terms should not be considered important in a single-labeled text classification process and should be filtered. However, algorithms such as *odds ratio, information gain, chi-squared, bi-normal separation* and *tficf* assign higher weight to these terms even if they appear in more than one category. We call these terms *ambiguous* terms.

To tackle this problem, we present a feature selection method called Ambiguity Measure (*AM*) (Mengle, Goharian & Platt, 2007) that assigns a high score to a term, if it appears consistently in only one specific category. The intuition is that the term that appears in only one category points stronger to that specific category and thus, is a better indicator in a single-labeled classification decision.

We apply *AM* on single labeled Naïve Bayes text classifier and compare *AM* with eight feature selection algorithms on five standard datasets from various subject domains, namely news feeds, web pages, and bio-medical text. Our results indicate that AM feature selection achieves statistically significant improvements on unbalanced datasets such as OHSUMED (20%) and Genomics (7.5%), and on balanced datasets such as WebKB (2.6%), 20NG (2.14%) and Reuters 21578 (0.25%) when compared to the best performing feature selection method out of the eight methods. However, the improvements on the unbalanced datasets are larger than the improvements on the balanced datasets.

Furthermore, we also explore the effects of the *AM* feature selection method when applied on single labeled *Support Vector Machine (SVM)* algorithm (Cortes & Vapnik, 1995; Joachims, 1999; Yang, Zhang & Kisiel, 2003). *SVM* algorithm is one of the widely used text classification algorithms. Prior work (Joachims, 1998) indicates that *SVM* performs consistently better than Naïve Bayes, kNN, C4.5 and Rocchio text classifiers. However, one of the limitations of *SVM* is its training time complexity. (Yang, Zhang & Kisiel, 2003) shows that *SVM* has a higher time complexity for training a model than other text classification algorithms. To overcome this limitation of *SVM*, feature selection methods are used as a preprocessing step before training *SVM* (Wenqian et al., 2007; Novovicova & Malik, 2005; Yan et al., 2005). Many well-known feature selection algorithms are used with *SVM* to improve its accuracy and efficiency. We use the *AM* feature selection method as a pre-processing step for the Support Vector Machine classifier (Mengle & Goharian, 2008). The features whose *AM* scores are below a given threshold, i.e., more ambiguous terms, are purged while the features whose *AM* scores are above a given threshold are used for the *SVM* learning phase. We favorably compare the results of *AM* feature selection algorithm with the same eight feature selection algorithms reported in (Wenqian et al., 2007; Yan et al., 2005) on four of the standard benchmark datasets. We also empirically show that using *AM* feature selection with *SVM* reduces the training time by more than 50%, while maintaining the accuracy of the classifier.

Prior Work

Various techniques are used for finding an "optimal" subset of features from a larger set of possible features. Exhaustively trying all the subsets is not computationally feasible. Hence, automatic feature selection algorithms are used to find the most important features in the feature set. In this section, we present the commonly used feature selection algorithms.

Odds Ratio

The basic idea of using *odds ratio* (Mladenić & Grobelnik, 1998) is to calculate the odds of a term occurring in the positive class (the category a term is related to) normalized by the odds of that term occurring in the negative class (the category a term is not related to). The *odds ratio* of a term t_k for a category c_i is defined using Equation 1:

$$Odds\ Ratio\ (t_k, c_i) = \frac{P(t_k \mid c_i)[1 - P(t_k \mid \overline{c_i})]}{[1 - P(t_k \mid c_i)]P(t_k \mid \overline{c_i})} \qquad ..1$$

Odds Ratio is known to work well with the Naive Bayes text classifier algorithm (Mladenić et al., 2004).

Information Gain

Information gain (Quinlan J., 1986) is commonly used as a surrogate for approximating a conditional distribution for text classification. In information gain, class membership and the presence/absence of a particular term in a given category are seen as random variables; one computes how much information about the class membership is gained by knowing the presence/absence statistics. If the class membership is interpreted as a random variable C with two values, positive (c) and negative ($\bar{c}$), and a word is likewise seen as a random variable T with two values, present (t) and absent ($\bar{t}$), then *Information Gain* is defined as Equation 2:

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

Chi-Squared

χ^2 test is used in statistics to test the independence between two events. In text classification, χ^2 (Yang & Pedersen, 1997; Galavotti & Sebastiani, 2000; Wu & Flach, 2001) is used to measure the association between a category and features. The χ^2measure of a term t_k for a category c_i is defined using Equation 3:

$$\chi^2(t_k,c_i) = \frac{P(t_k \mid c_i)P(\overline{t_k} \mid \overline{c_i}) - P(t_k \mid \overline{c_i})P(\overline{t_k} \mid c_i)}{\sqrt{P(t_k)P(\overline{t_k})P(c_i)P(\overline{c_i})}} \qquad ..3$$

Thus, $\chi^2(t_k, c_i)$ score indicates the weight of term t_k with respect to category c_i. If a term is close to more categories, then the score of that term is higher. The score of each term t_k is calculated using Equation 4:

$$\chi^2(t_k) = \sum_{i=1}^{c} P(c_i)\chi^2(t_k,c_i) \qquad ..4$$

Bi-Normal Separation

In *Bi-Normal Separation (BNS)* feature selection method (Forman, 2003; Forman, 2008), the occurrence of a given term is modeled in each document by a random normal variable that exceeds a hypothetical threshold. The prevalence rate is calculated with respect to both positive and negative classes. Prevalence rate can be defined as the area under the curve past the certain threshold. Thus, if a term consistently appears in positive class, the threshold is farther from the tail of the curve than that of the negative class. BNS is calculated based on the separation between these two thresholds. Thus, if a term appears more consistently in positive class than the negative class, it is assigned a higher BNS score. BNS is calculated using the Equation 5.

$$BNS = F^{-1}\left(\frac{tp}{tp+fn}\right) \;-\; F^{-1}\left(\frac{fp}{fp+tn}\right) \qquad ..5$$

Where, tp is the number of positive cases containing the word; fp is the number of negative cases containing the word; fn is the number of positive cases that do not contain the word; tn is the number of negative cases that do not contain the word, and F^{-1} is the standard normal distribution's inverse cumulative probability function. As reported in (Forman, 2003), BNS+F1 yields the best performance on most of the tasks in comparison with *odds ratio*, *information gain* and *Chi-Squared*.

F1 metrics (Equation 6) is the harmonic mean of precision (Equation 7) and recall (Equation 8).

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \qquad ..6$$

$$Precision = \frac{tp}{tp + fp} \qquad ..7$$

$$Recall = \frac{tp}{tp + fn} \qquad ..8$$

Improved Gini Index

In *Gini Index* (Breiman, Friedman and Olshen, 1984), if a term appears in every document of class c_i, then it receives a high *Gini Index* score. (This is regardless of term occurrence in other classes.) When a term is distributed evenly in the documents of various categories, the term is then assigned a lower *Gini Index* score. Gini Index for a term t_k can be calculated using Equation 9.

$$Gini(t_k) = P(t_k)\left(1 - \sum_i P(c_i \mid t_k)\right) + P(\overline{t_k})\left(1 - \sum_i P(c_i \mid \overline{t_k})\right) \quad ..9$$

However, *Gini Index* fails to consider the frequency of documents where the term occurs within larger categories. The categories are generally unbalanced with respect to the number of training documents. Hence, the *Gini Index* score is biased with respect to categories that have large number of training documents. (Wenqian et al., 2007) constructs a new function called *Improved Gini Index* that considers a term's condition probability and combines the posterior probability and condition probability to avoid the effects of unbalanced classes in datasets. *Improved Gini Index* of a given term t_k is defined using Equation 10:

$$I - Gini\ (t_k) = \sum_{i=1}^{C} P(t_k \mid c_i)^2\, P(c_i \mid t_k)^2 \qquad ..10$$

Tficf

In *tficf* (Chih & Kulathuramaiyer, 2004), *tf* refers to term frequency of a term in a given category and *icf* refers to inverse category frequency, i.e., the ratio of total number of categories in a dataset to the number of categories a term falls into. *tficf* scheme does not discriminate between terms that occur frequently in a small subset of documents in a category and terms that are present in a large number of documents throughout a category. Thus, *tficf* considers that the lesser a term occurs across categories, the higher is its score. *tficf* of a term t_k in category c_i is defined using Equation 11:

$$tficf\ (t_k, c_i) = tf\ (t_k, c_i) \log\left(\frac{|C|}{cf\ (t_k)}\right) \qquad ..11$$

Where $|C|$ refers to the total number of categories in a dataset, $tf(t_k,c_i)$ is the term frequency of a term t_k in category c_i and $cf(t_k)$ refers to the number of categories in which a term t_k appears.

Tfidf

In *tfidf* (Chih & Kulathuramaiyer, 2004), *tf* refers to term frequency of a term in a given document. *idf* is defined as the inverse document frequency, i.e., the ratio of the total number of documents present in a dataset to the number of documents a given term appears in. Higher *idf* of a term indicates that the term appears in relatively few documents and may be more important during the process of text classification. *tfidf* is a commonly used technique for term weighing in the field of information retrieval (Grossman & Frieder, 2004) and is also used in text classification (Lavelli, Sebastiani & Zanoli, 2004; Debole & Sebastiani, 2003). *tfidf* of a term t_k in document d_i is defined using Equation 12:

$$tfidf\ (t_k, d_i) = tf\ (t_k, d_i) \log\left(\frac{|D|}{df\,(t_k)}\right) \qquad ..12$$

Where $|D|$ refers to the total number of documents in a dataset, $tf(t_k,d_i)$ is the term frequency of a term t_k in document d_i and $df(t_k)$ refers to the number of documents in which term t_k appears.

Orthogonal Centroid Feature Selection (OCFS)

The Orthogonal Centroid Feature Selection (OCFS) (Yan et al., 2005) selects features optimally according to the function implied by the Orthogonal Centroid algorithm. The centroid of each class (m_j) and also for the entire dataset (m) is calculated using training data. A score for term t_k is calculated using Equation 13.

$$OCFS\ (t_k) = \sum_{j=1}^{c} \frac{n_j}{n} (m_j^{t_k} - m^{t_k})^2 \qquad ..13$$

Where n_j is the number of training samples that belong to category j and n is the total number of training samples. The feature set is pruned by selecting only the features whose scores are higher than a threshold. OCFS is not greedy in nature like *odds ratio* or *information gain*. Hence, the OCFS algorithm can be optimized based on the objective function that is implied by the Orthogonal Centroid algorithm and has been shown to improve over traditional algorithms.

Methodology

Initially, we describe our motivation behind AM feature selection algorithm and formally define AM. Secondly, we discuss the differences between AM measure and various feature selection algorithms. Finally we define a methodology for using feature selection algorithms with text classification algorithms such as NB and SVM.

AM Feature Selection Algorithm

Initially, we describe the intuitive motivation behind our *AM* feature selection approach and then provide a formal definition. We consider the human perception of the topic of a document by glancing at the document and capturing its *keywords*. Instead of using all the terms in a document to determine the subject of a text, normally one bases his/her decision on the most unambiguous words that the eye captures. The person then has an idea of the topic of the document. Some words can easily suggest the category in which the document can fall into. For example, if the document has phrases like *Chicago White Sox* and *MLB World Series Champion*, then one can suggest that the document relates to baseball in particular and sports in general. The sample text below is taken from Wikipedia[1]. By having a glance at this text, the reader can guess the category of this text.

"Metallica is a Grammy Award-winning American heavy metal/thrash metal band formed in 1981 and has become one of the most commercially successful musical acts of recent decades. They are considered one of the "Big Four" pioneers of thrash metal, along with Anthrax, Slayer, and Mega-death. Metallica has sold more than 90 million records worldwide, including 57 million albums in the United States alone."

The text seems to be about *Music*. Our human perception is based on our knowledge of the domain or what we hear or read on various subjects in daily life. Thus, without reading this specific text completely, one can confidently claim that the text belongs to *Music* rather than *Terrorism* or *Politics.*

Some terms may be stronger indicators that a given text belongs to a certain category than to others. Thus, we can give a score as to how strongly a term suggests a particular category. We clarify this by giving the following hypothetical example.

"Carolina Panthers lost the Superbowl title to Chicago Bears due to the final minute touchdown."

In the above sentence, we have terms such as *Bears* and *Panthers*, which are related to *wildlife.* On the other hand, they are also the names of famous NFL football teams. Here we notice uncertainty in classifying the text to *Wildlife* or to *Sports* categories. The terms such as *Superbowl* and *touchdown*, in the same given text, suggests more certainty that the text is about *Sports.*

We define an *Ambiguity Measure, AM,* for each term t_k with respect to category c_i, using Equation 14. The

[1] Wikipedia. http://en.wikipedia.org/wiki/Metallica

Table 1. Ambiguity Measure (*AM*) example

Term	**Metallica**		**Anthrax**		**Records**	
Category	**Count**	***AM***	**Count**	***AM***	**Count**	***AM***
Medicine	0	0.00	800	0.80	150	0.15
Music	990	0.99	150	0.15	240	0.24
Sports	10	0.01	00	0.00	330	0.33
Politics	0	0.00	50	0.05	280	0.28

maximum *AM* score for term t_k with respect to all categories is assigned as *AM* score of term t_k (Equation 15).

$$AM(t_k, c_i) = \left(\frac{tf(t_k, c_i)}{tf(t_k)} \right) \quad ..14$$

$$AM(t_k) = \max\ (AM(t_k, c_i)) \quad ..15$$

Where $tf(t_k,c_i)$ is the term frequency of a term t_k in category c_i and $tf(t_k)$ is the term frequency of a term t_k in the entire collection.

We then assign a higher score to unambiguous terms. In the above example, the term *touchdown* has a higher *AM* than that of the terms *Bears* and *Panthers*. The *AM* score is close to 1 if the term is unambiguous. Conversely, if *AM* is closer to 0, the term is considered more ambiguous and may point to more than one category.

The *AM* score for the feature *Metallica*, for the sample text, is 0.99, which indicates that the feature *Metallica* is an unambiguous feature and should be kept and not filtered (Table 1). *Anthrax* is related to the *Medicine* category with an *AM* score of 0.80. *Anthrax* is also the name of a famous music band in 1980s. Hence, it also appears in the category *Music*. Thus, the *AM* of *Anthrax* is less than *Metallica*. In some cases the *AM* score of some features is low as they appear consistently in multiple categories. Example of such is the term *Records*, which may appear in all different categories. Thus, the *AM* score of such term is low (0.33), and it is desirable to filter out such features. This reduction in dimensionality of the feature set increases the accuracy by avoiding the terms that have lower *AM* scores. We empirically determine a threshold and filter out features whose *AM* scores are below that given threshold.

Differences

The feature selection methods of *odds ratio, information gain, BNS+F1* and *Chi-Squared* assign a high score to a term even if it appears in more than one category. Using such features do not assist a single labeled text classifier to distinguish between categories. *AM* feature selection method assigns a high score to a term, if it appears consistently in only one specific category. Such terms then can point the classifier to that specific category. For example, consider a term t_1 with half of its occurrences in one category c_1 and distributed uniformly across the other categories. Term t_1 confidently points to category c_1 and hence is assigned *AM* score of 0.50. Consider another term t_2 with 49% of occurrences in category c_1 and the other 51% of occurrences concentrated in two other categories c_2 and c_3. An *AM* score of 0.49 is assigned to term t_2. As our goal is single-labeled classification, *AM* assigns a higher score to term t_1 than term t_2 as it points more confidently to category c_1. However, algorithms such as *information gain, odds ratio, BNS+F1* and *Chi-Squared* assign a score to a term that is inversely proportional to the number of categories that term appears in. Hence, term t_2 (occurs in three categories) is assigned a higher score than t_1 (occurs in all categories). However, the term t_2 may mislead a single-labeled classifier as it also points to categories c_2 and c_3 each with a lower probability (25.5%). Term t_1 confidently points to only category c_1 and hence, should be assigned a higher score than t_2.

In the *Improved Gini Index* method, the probabilities of a term with respect to all the categories are considered. If the term t_k appears in many documents of category c_i, then t_k is assigned a high score. In a situation where term frequency of the term t_k in categories c_i and c_j is the same, and also it appears in every document of both categories c_i and c_j, then t_k is assigned a high score. However, as term t_k belongs to two different categories it is ambiguous. Our proposed *AM* feature selection method avoids such situation and assigns a low score to features like t_k.

Using *tfidf* and *tficf* methods, the terms that appear with a low frequency in only a single category are purged during the feature selection process. However, such terms are unambiguous and point to a single category. Another problem is that some terms have a similar distribution in more than one category (low *idf* or *icf*), but have a high term frequency. These terms are selected during the process of feature selection as the term frequency of such terms is high. These terms are ambiguous, as they do not point strongly to only a single category. The *AM* feature selection method avoids such situations by only considering the ratio between the numbers of occurrences of a term in a given category to the total number of occurrences of that term in the training set. Thus, both these situations are avoided.

In *OCFS* the training and the testing time is quadratic as the centroids of each class and the entire dataset are calculated. However, *AM* feature selection method trains and tests in linear time (is discussed later in the paper).

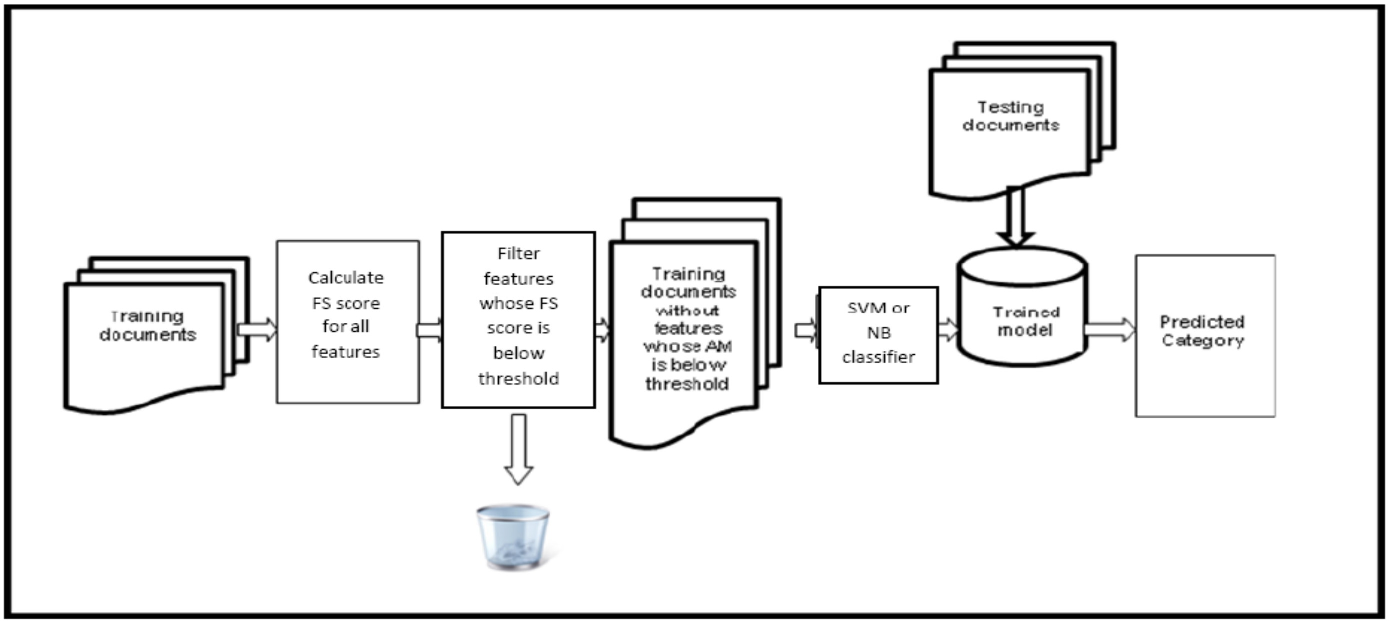

Training documents
Calculate FS score for all features
Filter features whose FS score is below threshold
Training documents without features whose AM is below threshold
SVM or NB classifier
Testing documents
Trained model
Predicted Category

Figure 1. Block diagram for using feature selection method on a text classifier

Using feature selection algorithms on SVM and NB text classifiers

We evaluate our feature selection algorithm on SVM and Naïve Bayes text classifiers. SVM is commonly used as was shown to perform better in terms of effectiveness than other text classifiers such as Naïve Bayes, kNN, C4.5 and Rocchio (Joachims, 1998). Naïve Bayes algorithm is, however, more efficient and scalable than other algorithms (Yang, Zhang & Kisiel, 2003).

We present the methodology for applying feature selection algorithms on SVM and NB text classifiers (Figure 1). This process is divided into four phases.

Phase 1. Calculating feature selection scores

In the pre-processing step, *feature selection score* for each feature in training documents is calculated.

Phase 2. Filtering terms with lower feature scores

We only keep the features in training documents if the *feature score* of a term is above a certain empirically determined threshold. We determine these thresholds by exhaustively optimizing the results of each algorithm on the testing documents. The choice of testing versus separate validation set is to be consistent with the prior works (Wenqian, et al, 2007) (Yan, et al., 2005) (Chih, Kulathuramaiyer, 2004) that we compare our work with.

We compare AM with both local and global feature selection algorithms. We globalize the local feature selection algorithms by selecting the terms with the highest local scores. Additionally, we also experiment using round robin method (Forman, 2004) to convert local feature selection scores into global scores.

Phase 3. Training the text classifier

Pruned documents from *Phase 2* are used by NB and SVM classifiers to train a text classification model. For NB, we use the traditional NB classifier as explained in (Mccullum and Nigam, 1998) to create a text classification model. We use the linear *SVM* kernel, as the non-linear versions gain very little in terms of performance (Mladenić et al., 2004). For training and testing the *SVM* model, we use LibSVM 2.84[2] software that is commonly used for classifying the documents into binary or multi-labeled categories.

Phase 4. Classifying documents

In the testing phase, the trained text classification model is used to classify testing documents by predicting a category for each. Unlike the traditional Naïve Bayes text classifier, we as in (Rennie, Teevan & Karger, 2003) do not consider prior probability while predicting the category for a testing document. As SVM only classifies documents into two classes (binary classifier), we use one against all (Yi & Zheng, 2005) technique to run SVM on multiclass datasets.

We use single labeled classification in this work to classify documents. Hence, only one category is predicted for each testing document by the text classifier.

[2] Chang C.C., Lin C.J., LIBSVM: a library for support vector machines, 2001.

Table 2. Time and space complexity for applying AM on Naive Bayes and SVM

Classifier	Training time	Testing time per document	Space Complexity
Naive Bayes using *AM*	$O(N L_d + M V)$	$O(M L_v)$	$O(M V)$
SVM using AM	$O(N L_d + M V + M N^c)$ $c \approx 1.2 \sim 1.5$	$O(M L_v)$	$O(M V + N L_v + q^2)$

N - number of training documents

L_d- average document length

M- number of categories

L_v - average number of unique terms in document

V – size of vocabulary (features)

q – constant that depends on the iterations needed

Time and Space Complexity Analysis

AM scores are computed in linear time as training documents arrive. However, the scalability of using AM depends on the text classifier. The comparison of time and space complexity for applying AM on Naive Bayes and SVM are given in Table 2, and are discussed in the subsections that follow.

Analysis of time complexity for applying AM on Naïve Bayes

The term frequency of each term per category is calculated. Thus, *Naïve Bayes* parses NL_d terms during the training phase. For every term in the vocabulary, *M* different AM scores are calculated which takes O(MV) time. Thus, the training time for Naive Bayes using *AM* is also $O(NL_d+MV)$ and equates to $O(NL_d)$(as $MV<<NL_d$). During the testing phase, we calculate the product of *AM* of terms present in the testing document with respect to each category, which takes $O(M L_v)$.

A lexicon of all the terms in vocabulary (*V*) and their *AM* scores with respect to all *M* categories are stored as the NB model. Many of the features are filtered during feature selection process, thus only some of the features and their *AM* scores are stored. Space needed by *Naïve Bayes* using *AM* is O(MV).

Analysis of time complexity for applying AM on SVM

As shown, the training time for Naive Bayes using *AM* is $O(NL_d+MV)$. Thus, *AM* for all the features in training set can be found in linear time. SVM, however, trains in quadratic time. Algorithms used in LibSVM train in $O(M N^c)$ where $c \approx 1.2 \sim 1.5$ (Yang, Zhang & Kisiel, 2003). Hence, the total time taken for training a model using *AM* as a preprocessing step of SVM is $O(NL_d+MV+MN^c)$. However, as NL_d and MV are much smaller than MN^c, we consider the training and testing time for using *AM* with SVM as $O (M N^c)$.

The space taken for storing SVM model is $O(N L_v + q^2)$ where q is a constant that depends on the iterations needed. Hence, the space complexity for using AM with SVM is $O(M V + N L_v + q^2)$.

Experimental Setup

We empirically evaluated the effectiveness of *AM* feature selection algorithm using five benchmark data sets (Reuters 21578, 20 Newsgroups, WebKB, OHSUMED, Genomics), which are commonly used in text classification evaluation. The details on these data sets are given in Table 3. We intentionally chose these datasets, which consist of news articles, web pages and bio-medical documents, to show the effects of AM on different domains. Although we observe different accuracies across different domains, AM consistently outperforms other feature selection algorithms over all domains. To show the scalability of our AM feature selection approach, using NB classifier, we also show the effectiveness and efficiency analysis on TREC 2005 Genomics dataset, which contains 4.5 million documents. We do not show the results for TREC Genomics 05 on SVM classifier, as SVM is not scalable to be used on very large datasets. (The training time for a SVM model for TREC 05 Genomics is almost 4 days.)

In all our experiments, we use a single computer, with an AMD Athlon 2.16Ghz processor and 1 GB of RAM. A brief explanation about the benchmark datasets that are used in our experiments is given below.

Reuters- 21578 Dataset

Reuters 21578[3] corpus contains the Reuters news articles from 1987. These documents range from multi-labeled, single-labeled, or not labeled. The average document length in Reuters 21578 dataset is 200 (non-unique) terms per document. Reuters dataset consists of a total number of 135 categories (labels), ten of which have significantly more documents than the rest of the categories. Thus, commonly the top 10 categories are used to evaluate the accuracy of the classification results. The top 10 categories of Reuters 21578 are “earn”, “acq”, “money-fx”, “grain”, “trade”, “crude”, “interest”, “wheat”, “corn” and “ship”.

[3] Lewis D., Reuters-21578, http://www.daviddlewis.com/resources/testcollections/reuters21578.

Table 3. Benchmark datasets used for our experiments

Datasets	No. of documents	No. of Categories	Size of dataset	Domain
Reuters 21578	21,578	Top 10 categories	28 MB	News Articles
20 News Group	20,000	20 categories	61 MB	News Articles
WebKB	8,282	7 categories	43 MB	Web Pages (University websites)
OHSUMED	54,710(Total) 39,320 (Subset)	Top 50 categories	382 MB	Bio-medical Documents
GENOMICS (TREC 05)	4.5 million (Total) 591,689 (Subset)	Top 50 categories	15.5 GB	Bio-medical Documents

20 Newsgroup (20NG) Dataset

20 Newsgroup[4] (20NG) consists of total of 20,000 documents that are categorized into twenty different categories. Each category contains 1,000 documents. The average document length in 20NG dataset is 311 terms per document. Thus, the average size of the documents is much larger than those in Reuters 21578 dataset. Some of the newsgroups categories are very closely related to each other (e.g., comp.sys.ibm.pc.hardware and comp.sys.mac.-hardware), while others are highly unrelated (e.g., misc.forsale and soc.religion.christian). This characteristic contributes to the difficulty of categorization of documents that belong to very similar categories.

WebKB Dataset

WebKB dataset[5] is a collection of web pages from four different college websites namely Cornell, Texas, Washington, Wisconsin and some miscellaneous web pages. These web pages are pre-classified into seven categories: student, faculty, staff, department, course, project and others. WebKB contains 8,282 web pages. The average document length in WebKB dataset is 130 terms.

OHSUMED Dataset

OHSUMED (Hersh, Buckley, Leone & Hickman, 1994) is a collection of Medline documents, i.e., medical citations, from 1987 to 1991, and is commonly used for bio-medical literature search evaluation and classification. We use only the top (largest) 50 categories with documents published in 1987. The average document length in OHSUMED dataset is 63 terms per document. The distribution of documents in OHSUMED dataset is uneven. The largest category contains 2,415 documents, while the smallest category contains 873 documents. Hence, more training data are available for some categories as compared to others.

TREC 2005 Genomics Dataset

TREC 05 GENOMICS is a collection of 4.5 million biomedical documents and is 15.5 GB in size. This is the largest publicly available benchmark dataset that contains categorized (labeled) documents in the domain of bioinformatics. The average document length is 183 terms per document.

We are not aware of any text classification efforts on TREC 05 GENOMICS[6] data set. Thus, for this dataset no comparison with prior efforts was possible. We used the data processed by (Urbain, Goharian & Frieder, 2007). They use a pre-processing model that breaks up gene names and is shown to perform well. Acronyms and their long-forms are identified during preprocessing using the Schwartz and Hearst algorithm (Schwartz & Hearst, 2003). An example of such long-short form would include “immuno deficiency enzyme (IDE)”, and a short-long form would include “IDE (immuno deficiency enzyme)”. The algorithm works backwards through the long form text and attempts to identify corresponding letters in the acronym. All terms are tokenized, stop words removed, and lexical variants are generated. Porter stemming (Porter, 1997) is used on each token with the following exceptions: gene names (as defined by the Entrez Gene database); all upper case, mixed case, alpha-numeric terms; and non-gene terms that would become a gene name after being stemmed. Similar to OHSUMED dataset, the top (largest) 50

[4] Lang K., Original 20 Newsgroups Dataset. http:// people.csai.mit.edu/jrennie/20Newsgroups

[5] WebKB dataset. http://www.cs.cmu.edu/afs/ cs.cmu.edu/ project/theo-20/www/data/

[6] TREC 2005 Genomics dataset. http://ir.ohsu.edu/genomics/

categories are chosen that contain highest number of documents for GENOMICS dataset. Similarly, the categories are ranked based on the number of documents. This subset of Genomics dataset contains 591,589 documents. The category that contains the highest number of documents contains 295,773 documents while the category among top 50 categories that contains least number of documents has 8,049 documents. Hence, if we choose the categories after the top 50, then the number of training documents in these categories is very low, leading to a lower classification accuracy.

Evaluation Metrics

To evaluate the effectiveness of our approach and compare to the state of the art feature selection research results, we use the commonly used evaluation metrics precision, recall and F1 measure.

$$\text{Precision (P)} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \qquad \text{.. 16}$$

Precision (Equation 16) is defined as the ratio of correct classification of documents into categories to the total number of attempted classifications.

$$\text{Recall (R)} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \qquad \text{.. 17}$$

Recall (Equation 17) is defined as the ratio of correct classifications of documents into categories to the total number of labeled data in the testing set.

$$\text{F1 Measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \qquad \text{..18}$$

F1 measure (Equation 18) is defined as the harmonic mean of precision and recall. Hence, a good classifier is assumed to have a high F1 measure, which indicates that classifier performs well with respect to both precision and recall.

We present the micro-averaged results for precision, recall and F1 measure. Micro-averaging considers the sum of all the true positives, false positives and false negatives that are generated in ten runs of 10-fold cross validation (Lewis, 1991).

Results

We organize the results into two subsections. In the first subsection, we present the result for Naïve Bayes classifier using *AM* feature selection method. In the second subsection, the results for *AM* feature selection with SVM classifier are presented.

Naive Bayes using AM

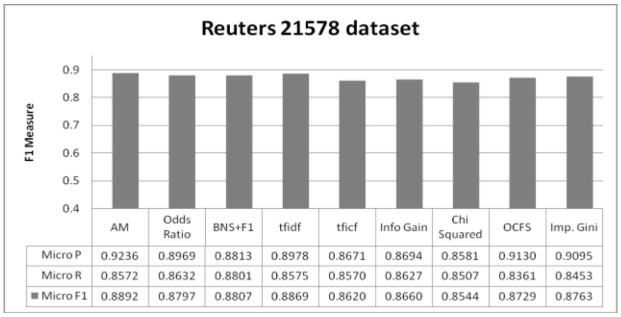
We evaluated the experimental results using Reuters 21578, 20NG, WebKB, OHSUMED and TREC 05 Genomics datasets. We present the comparison of *AM* feature selection algorithm with the eight feature selection algorithms explained earlier in the prior work section. We varied the threshold to identify the optimal F1 measure for each feature selection method. The results show that *AM* outperforms the others statistically significantly with a confidence level of at least 95%. We demonstrate the effects of using round robin method on AM that is used for globalizing the localized feature selection score. We also present the effects of *AM* on the training and testing time for Naive Bayes classifier.

Comparison with other feature selection algorithms using Naive Bayes classifier

We used stratified 10-fold cross validation for all the datasets except WebKB. We used a standard 4-1 split for WebKB where the data for three universities are used for training and the data for one university is used as a testing set. We varied thresholds to observe the best results with respect to F1. Our results show that AM comparatively performs better than the next best performing feature selection algorithms by 20%, 7.5%, 0.25%, 2.14%, and 2.6%, on OHSUMED, TREC 05 Genomics, Reuters 21578, 20 Newsgroups, WebKB datasets, respectively.

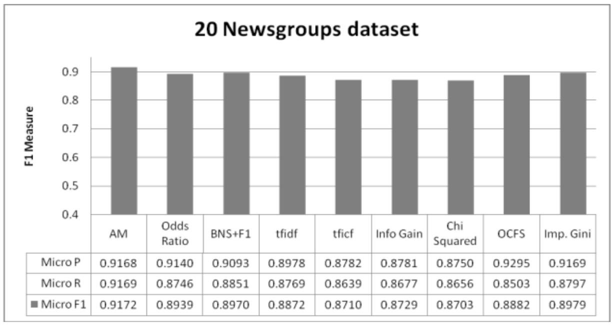
Figure 2 shows the comparison of eight feature selection algorithms on Reuters 21578 dataset with respect to F1 measure. Our experimental results show that *AM* (Precision: 92.36%, Recall: 85.72%, F1: 88.92%) performs better than *tfidf* (Precision: 90.78%, Recall: 86.69%, F1: 88.69%) and *BNS+F1* (Precision: 88.13%, Recall: 88.01%, F1: 88.07%) that are the next best performing algorithms. As all the feature selection algorithms perform well on Reuters 21578 dataset, the F1 improvement when using AM measure is only 0.25% (95% confidence). Statistical significance of the *AM* with respect to other feature selection algorithms for various datasets is reported in Table 4. For 20 Newsgroups (Figure 3), *AM* (Precision: 91.68%, Recall: 91.69%, F1: 91.72%) performs significantly better than the next best feature selection algorithm, *improved gini index* (Precision: 91.69%, Recall: 87.97%, F1: 89.79%), by 2.14%. Although the improvement is marginal, the results are statistically significant by at least 95% confidence.

The results on WebKB dataset, which are given in Figure 4, show that *AM* (Precision: 74.34%, Recall: 73.76%, F1: 74.05%) performs better than the second best performing algorithm, *improved gini index* (Precision: 71.74%, Recall: 72.56%, F1: 72.15%), by 2.6%. WebKB dataset consists of web pages, which contains images, tables and other anchor text. Classifying such documents is more difficult than



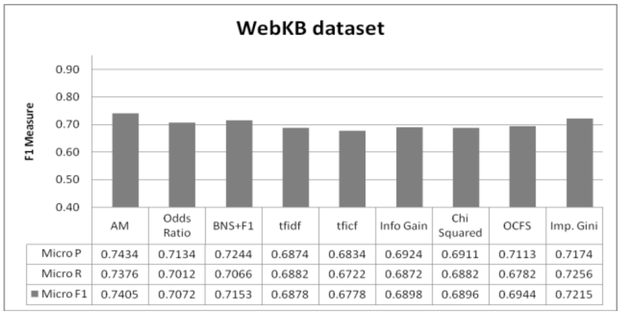
	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.9236	0.8969	0.8813	0.8978	0.8671	0.8694	0.8581	0.9130	0.9095
Micro R	0.8572	0.8632	0.8801	0.8575	0.8570	0.8627	0.8507	0.8361	0.8453
■ Micro F1	0.8892	0.8797	0.8807	0.8869	0.8620	0.8660	0.8544	0.8729	0.8763

Figure 2. Comparison of *AM* with other feature selection methods in terms of F1 measure on Reuters 21578 dataset for Naive Bayes



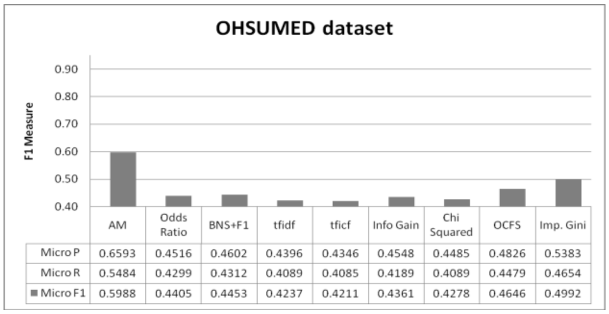
	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.9168	0.9140	0.9093	0.8978	0.8782	0.8781	0.8750	0.9295	0.9169
Micro R	0.9169	0.8746	0.8851	0.8769	0.8639	0.8677	0.8656	0.8503	0.8797
■ Micro F1	0.9172	0.8939	0.8970	0.8872	0.8710	0.8729	0.8703	0.8882	0.8979

Figure 3. Comparison of *AM* with other feature selection methods in terms of F1 measure on 20 Newsgroups dataset for Naive Bayes



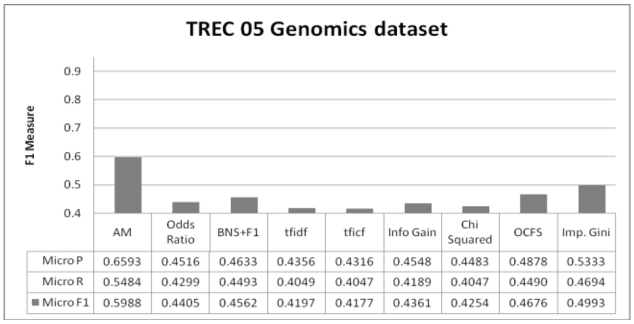
	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.7434	0.7134	0.7244	0.6874	0.6834	0.6924	0.6911	0.7113	0.7174
Micro R	0.7376	0.7012	0.7066	0.6882	0.6722	0.6872	0.6882	0.6782	0.7256
■ Micro F1	0.7405	0.7072	0.7153	0.6878	0.6778	0.6898	0.6896	0.6944	0.7215

Figure 4. Comparison of *AM* with other feature selection methods in terms of F1 measure on WebKB dataset for Naive Bayes



	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.6593	0.4516	0.4602	0.4396	0.4346	0.4548	0.4485	0.4826	0.5383
Micro R	0.5484	0.4299	0.4312	0.4089	0.4085	0.4189	0.4089	0.4479	0.4654
■ Micro F1	0.5988	0.4405	0.4453	0.4237	0.4211	0.4361	0.4278	0.4646	0.4992

Figure 5. Comparison of *AM* with other feature selection methods in terms of F1 measure on OHSUMED dataset for Naive Bayes



	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.6593	0.4516	0.4633	0.4356	0.4316	0.4548	0.4483	0.4878	0.5333
Micro R	0.5484	0.4299	0.4493	0.4049	0.4047	0.4189	0.4047	0.4490	0.4694
■ Micro F1	0.5988	0.4405	0.4562	0.4197	0.4177	0.4361	0.4254	0.4676	0.4993

Figure 6. Comparison of *AM* with other feature selection methods in terms of F1 measure on TREC 05 Genomics dataset for Naïve Bayes

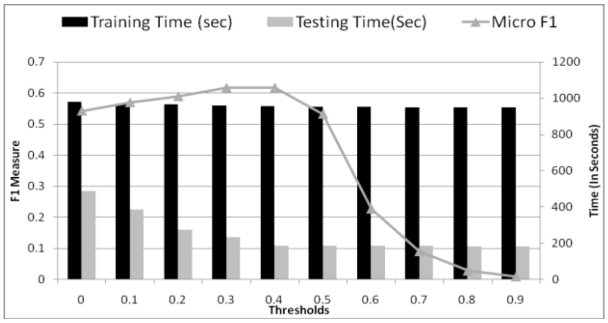


Figure 7. Effect of feature selection on training and testing time of Naïve Bayes using AM on TREC 05 Genomics dataset

Table 4. Statistical comparison of AM and other feature selection algorithms on Naïve Bayes with respect to F1 measure (Paired t-test)

Algorithm	Datasets				
	Reuters 21578	20 Newsgroups	WebKB	OHSUMED	Genomics
Odds Ratio	+	++	++	++	++
BNS+F1	+	++	++	++	++
tfidf	++	++	++	++	++
tficf	++	++	++	++	++
Info Gain	++	++	++	++	++
Chi-Squared	++	++	++	++	++
OCFS	++	+	++	++	++
Imp. Gini	+	+	++	++	++
+ : AM is statistically significantly better than the feature selection algorithm by 95% confidence ++ : AM is statistically significantly better than the feature selection algorithm by 99% confidence					

classifying plain documents from Reuters 21578 and 20 Newsgroups datasets. Hence, the classification effectiveness for WebKB dataset is lower than Reuters 21578 and 20 Newsgroups datasets.

On biomedical datasets, our results indicate that *AM* (Precision: 65.93%, Recall: 54.84%, F1: 59.88%) statistically significantly improves (20%) over *improved gini index* (Precision: 53.83%, Recall: 46.54%, F1: 49.92%) on OHSUMED dataset (Figure 5). *AM* (Precision: 61.71%, Recall: 60.54%, F1: 61.12%) also shows a statistically significant improvement of 7.5% over *improved gini index* (Precision: 61.71%, Recall: 52.64%, F1: 56.82%) for TREC Genomics 05 dataset (Figure 6). *Improved Gini index* is the second best performing algorithm on both these datasets.

Discussion

The motivation for using AM feature selection is to select terms that belong to only one category. As mentioned in introduction section, ambiguous features lead to wrong classification predictions in unbalanced datasets. Our results indicate that *AM* performs better than *odds ratio, information gain, tficf, tfidf, BNS+F1* and *Chi-Squared* on OHSUMED and Genomics datasets by more than 30% (comparative gain). OHSUMED and Genomics datasets are unbalanced and a large number of training documents belong to the top two categories. The feature selection methods such as *odds ratio, information gain, tficf, tfidf, BNS+F1* and *Chi-Squared* use both positive and negative examples to assign scores to the features. A high score is assigned to a feature even if it appears evenly in only two or three categories out of fifty. As the number of training documents in the top two or three categories is large, many features only appear in the top two or top three categories. Such features are assigned high scores. These features mislead the text classifier and hence, many false positives are generated during the testing phase. Such features are assigned a low AM score and are filtered during the process of feature selection.

Improved gini index nullifies the effects of unbalanced classes in dataset by combining the posterior probabilities and condition probabilities for each term. *OCFS* is optimized based on the number of documents available in each class. Hence, *improved gini index* and *OCFS* perform comparatively better than *odds ratio, information gain, tficf, tfidf, BNS+F1* and *Chi-Squared* algorithms. However, our results indicate that *improved gini index* and *OCFS* perform statistically significantly worse than *AM* on unbalanced dataset such as OHSUMED and Genomics.

Globalizing Feature Selection Scores

As feature selection algorithms such as *tficf, odds ratio, information gain, chi-squared, BNS+F1* and *AM* are local feature selection algorithms, we have used the traditional method (selecting the terms with the highest local scores) to convert their local scores to global feature selection score. Additionally, similar as in (Forman, 2004), we used round robin method to convert the local feature selection score into global score. Round robin method selects the top *n* features from each category. Thus, the categories with low number of training documents also have the same number of features in the feature set that represent them. This method improves the effectiveness in identifying the documents that belong to categories that have less training documents and leads to an improvement in macro-F1, which is the average of F1 measure of all categories. However, the classification accuracy of the categories with a large number of training documents decreases. As we are using stratified splits for each dataset, the number of training documents that belong to a category is directly related to the number of testing documents that belong to that category. Hence, the micro-F1 measure when using round robin method decreases. We provide the results of AM versus using AM with round robin method in Figure 8. We observed that using round robin method improves macro F1 measure by 1.7% while decreasing the micro F1 by 5.2% for Reuters 21578 dataset.

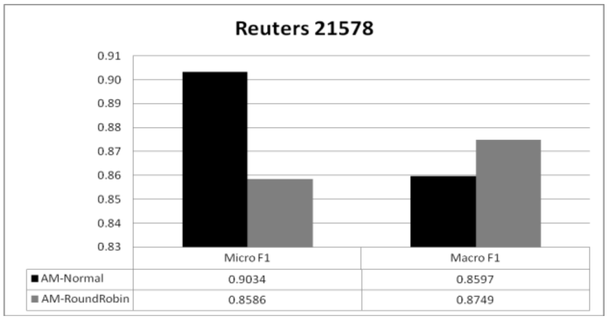
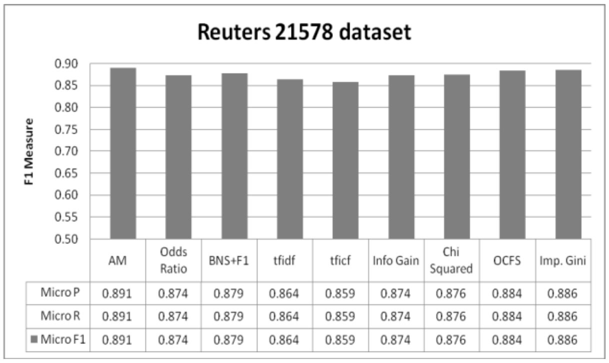


Figure 8. Comparison between AM with/without round robin method

Tradeoff of accuracy and time with respect to *AM* thresholds for Naive Bayes

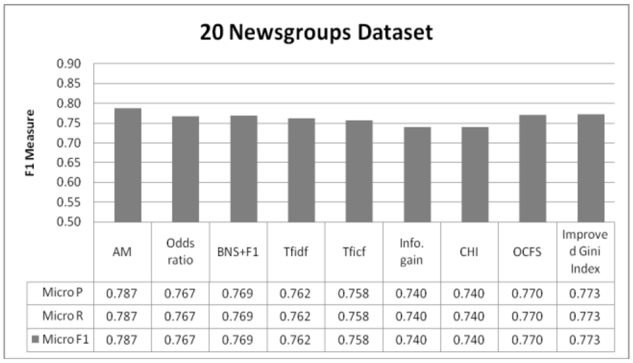
We now present the effects of *AM* threshold on the training and testing time of Naive Bayes using TREC 05 Genomics dataset (Figure 7). We performed similar experiments on other datasets and observed the same trends. As the TREC 05 Genomics dataset is relatively large, the trends with respect to training and testing time are observed clearly. Hence, we only report the results for TREC 05 Genomics dataset.

The training time complexity of Naive Bayes using *AM* is O(NL$_d$+MV) where *N* is the number of documents, L_d is the average document length, *M* is the number of categories and V is the total terms in the vocabulary. As *N*, L_d, *M* and *V* are all constant during the training phase, the training



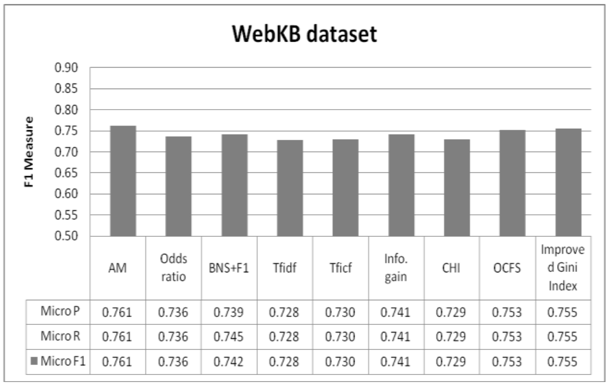
	AM	Odds Ratio	BNS+F1	tfidf	tficf	Info Gain	Chi Squared	OCFS	Imp. Gini
Micro P	0.891	0.874	0.879	0.864	0.859	0.874	0.876	0.884	0.886
Micro R	0.891	0.874	0.879	0.864	0.859	0.874	0.876	0.884	0.886
■ Micro F1	0.891	0.874	0.879	0.864	0.859	0.874	0.876	0.884	0.886

Figure 9. Comparison of *AM* with other feature selection methods in terms of F1 measure on Reuters 21578 dataset for SVM



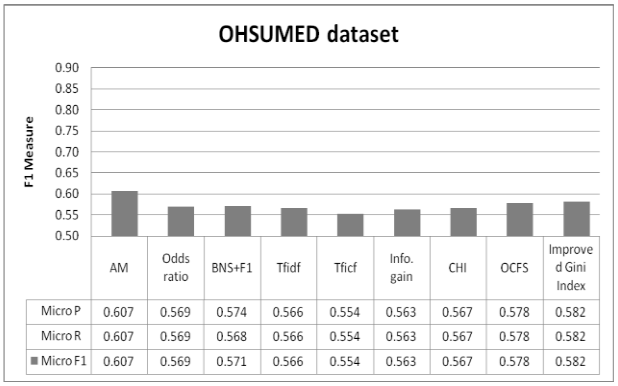
	AM	Odds ratio	BNS+F1	Tfidf	Tficf	Info. gain	CHI	OCFS	Improved Gini Index
Micro P	0.787	0.767	0.769	0.762	0.758	0.740	0.740	0.770	0.773
Micro R	0.787	0.767	0.769	0.762	0.758	0.740	0.740	0.770	0.773
■ Micro F1	0.787	0.767	0.769	0.762	0.758	0.740	0.740	0.770	0.773

Figure 10. Comparison of *AM* with other feature selection methods in terms of F1 measure on 20 Newsgroups dataset for SVM



	AM	Odds ratio	BNS+F1	Tfidf	Tficf	Info. gain	CHI	OCFS	Improved Gini Index
Micro P	0.761	0.736	0.739	0.728	0.730	0.741	0.729	0.753	0.755
Micro R	0.761	0.736	0.745	0.728	0.730	0.741	0.729	0.753	0.755
■ Micro F1	0.761	0.736	0.742	0.728	0.730	0.741	0.729	0.753	0.755

Figure 11. Comparison of *AM* with other feature selection methods in terms of F1 measure on WebKB dataset for our SVM



	AM	Odds ratio	BNS+F1	Tfidf	Tficf	Info. gain	CHI	OCFS	Improved Gini Index
Micro P	0.607	0.569	0.574	0.566	0.554	0.563	0.567	0.578	0.582
Micro R	0.607	0.569	0.568	0.566	0.554	0.563	0.567	0.578	0.582
■ Micro F1	0.607	0.569	0.571	0.566	0.554	0.563	0.567	0.578	0.582

Figure 12. Comparison of *AM* with other feature selection methods in terms of F1 measure on OHSUMED dataset for our SVM

time of our algorithm is constant (Figure 7). The features whose AM is above the threshold are kept. The space complexity of our Naive Bayes using AM is O(MV). As the size of *V* decreases when the threshold increases, there is a slight drop in the training time. Though there is marginal decrease in training time during feature selection phase, the time complexity for applying AM on Naive Bayes classifier is linear and is faster than other commonly used algorithms such as SVM.

The time complexity in the testing phase is O(ML$_v$), where L$_v$ is the total number of unique terms per testing document. As we start selecting fewer features (increase the threshold), the value of M remains constant, while the value of L$_v$ decreases. This is because fewer features are available in the feature set and hence, less unique terms are used from each document in testing set. Hence, as we increase the threshold the testing time consistently decreases. It is also observed that as the threshold increases up to 0.4, F1 measure increases while there is a reduction in testing time.

SVM using AM

In this section, we favorably compare our results of applying *AM* feature selection using SVM with the same eight feature selection algorithms. We varied the threshold to identify the optimal F1 measure for each feature selection method. We demonstrate how *AM* feature selection reduces the training time while improving the F1 measure. We also explain the effects of the *AM* threshold score on the classification results.

Comparison with other feature selection algorithms for SVM classifier

SVM trains with a time complexity of O(NL$_d$+MV+MN^c) where *N* is the number of documents, L_d is the average document length, *M* is the number of categories, V is the total terms in the vocabulary and c is a constant (c≈1.2~1.5). SVM in nature is not a scalable algorithm. We use the ModApte split for Reuters 21578 dataset and 9-1 split for 20 Newsgroups dataset and OHSUMED dataset as given on the LibSVM dataset

Table 5. Statistical comparison of AM and other feature selection algorithms with respect to F1 measure (Paired t-test)

Algorithm	Datasets				
	Reuters 21578	20 Newsgroups	WebKB	OHSUMED	Genomics
Odds Ratio	++	++	++	++	++
BNS+F1	++	++	++	++	++
tfidf	++	++	++	++	++
tficf	++	++	++	++	++
Info Gain	++	++	++	++	++
Chi-Squared	++	++	++	++	++
OCFS	++	++	++	++	++
Imp. Gini	++	++	++	++	++
++ : AM is statistically significantly better than the feature selection algorithm by 99% confidence					

website. We use a standard 4-1 split for WebKB where the data for 3 universities is used for training, and the data for one university is used as a testing set. We use these splits as they are readily available and commonly used in the prior works (Wenqian et al., 2007; Yan et al., 2005). AM performs statistically significantly better than the eight feature selection algorithms with a confidence of 99% (Table 5).

Improved Gini Index is the second best performing algorithm for all the four datasets. Therefore, we present the comparison of AM with *Improved Gini Index*. Our experimental results on Reuters 21578 (Figure 9) dataset indicate that AM (F1: 89.1%) performs better than *Improved Gini Index* (F1: 88.6%) by 0.56%. For 20 Newsgroups dataset (Figure 10), which is another dataset that contains news articles, AM (F1: 78.74%) outperforms improved gini index (F1: 77.3%) by 1.8%. The result on WebKB dataset (Figure 11), which is a dataset that contains web pages, indicates that *AM* (F1: 76.14%) outperforms *Improved Gini Index* (F1: 75.54%) by 0.8%. For the OHSUMED dataset (Figure 12) that contains biomedical documents, AM (F1: 60.74%) outperforms Improved Gini Index (F1: 58.23%) by 4.3%.

Discussion

Our results for SVM using AM also indicate that improvements in OHSUMED, which is very unbalanced dataset, are better than in other datasets. OHSUMED has the majority of documents in the first few (2-3) categories and fewer documents in the other 50 categories. This improvement is achieved due to the selection of the features that point to only one category (unambiguous features). SVM classification is based on the entire terms in the testing document and not on only unambiguous features. Hence, the improvements observed using SVM are smaller than those observed using Naïve Bayes classifier.

All features from the testing documents are used for classifying a document. LibSVM always predicts one category for each document. When a category is wrongly predicted, a false positive is generated; a false negative is also generated because a true prediction is not made. Precision and recall for all the runs using LibSVM are the same. Precision and recall vary for Naïve Bayes because when the *AM* threshold is high, the number of keywords is sparse and some documents do not contain any terms that are above the thresholds. Such documents are predicted as *uncertain* and only a false negative is generated in such cases. As we filter more features from the feature set, the number of *uncertain* cases increases and recall decreases.

Tradeoff of accuracy and time with respect to AM thresholds for SVM

We now report the effect of the *AM* threshold on F1 measure and the corresponding time taken to train the model and classify the documents using *SVM* classifier is depicted in Figure 13. Other datasets also show the same trends. The x-axis represents different threshold values and the y-axis represents micro-F1 measure and time. The threshold value indicates that all features whose scores are above that threshold are selected and the remaining features are filtered. As we apply *AM* feature selection, micro-F1 increases (Figure 13). We obtain the best micro-F1 when the threshold is set to 0.2. As the threshold is increased, the micro-F1 decreases. This indicates that when the threshold is less than 0.2, most of the features that are filtered are ambiguous and lead to a higher F1 measure. When the threshold is above 0.2, most of the features that are filtered contain relevant information. Thus, the F1 measure of the classifier decreases.

The training time includes the feature selection time and the time taken to train the *SVM* model. The *testing time* is the time taken by LibSVM to classify the testing data. Figure 13 demonstrates that when no feature selection is used i.e., when threshold is equal to zero, the time taken for training on OHSUMED dataset is 3356 seconds. When we reduce the dimensionality of feature set, by setting the threshold to 0.2, the training time also decreases to 1623 seconds. This shows that even though the learning time is reduced by more than 50%, we still obtain better F1 measure than when we do not apply any feature selection.

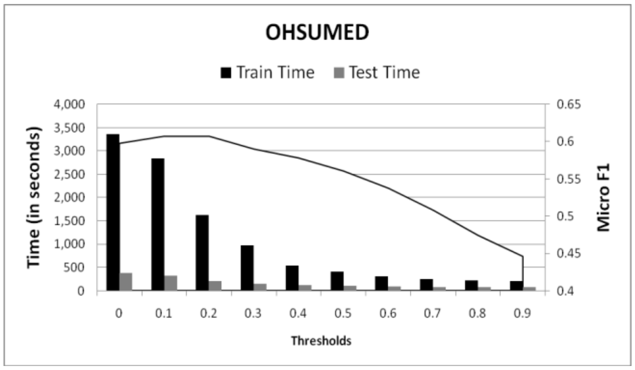


Figure 13. Effect of feature selection on training and testing time of Naïve Bayes using AM on OHSUMED dataset

One of the limitations of using feature selection algorithm on SVM is to find a proper threshold for a given dataset. We found the threshold for Reuters 21578 and WebKB dataset as 0.2 and for 20 Newsgroups and OHSUMED dataset as 0.3. To further investigate this problem, we experimented on two additional standard datasets from statlog collection (Michie, Spiegelhalter & Taylor, 1994) called DNA dataset (3 categories; 2,000 training documents; 1,186 testing documents) and Vehicle dataset (4 categories; 761 training documents; 85 testing documents). Similarly, we found that a threshold between 0.2 to 0.3 yields the best results on all the four datasets we used for our experimentations.

Conclusion

We presented a new feature selection algorithm called Ambiguity Measure (*AM*). The underlying premise behind the *AM* approach is the quick identification of unambiguous terms. We define unambiguous terms as features that belong to only one category. We showed how AM is used with Naïve Bayes classifier. The most unambiguous terms (keywords) from the training documents are selected using *AM* and a classification model is built. Based on this model, the documents that are to be classified are scanned to identify the keywords; and the ambiguity measures (*AM*) of the keywords are used to calculate the probability that the document falls in a specific category. The category with the highest probability is selected as the category for that document.

We empirically evaluated the performance of our methodology for using *AM* with Naive Bayes classifier using five standard benchmark data sets (Reuters 21578, 20 News Groups, WebKB, OHSUMED and TREC 05 Genomics collection). Our experimental results demonstrate that *AM* performs statistically significantly better than eight existing feature selection algorithms using five benchmark datasets with a confidence of at least 95%.

We also applied *AM* as a preprocessing step for *SVM* classification algorithm. We showed that *AM* feature selection reduces the training time of the *SVM* classifier, while maintaining its effectiveness. Experiments were performed on four standard benchmark datasets. Our results indicated that *AM* performs statistically significantly better than the current published state of the art feature selection algorithms on *SVM* classifier.

Our results also indicated that AM feature selection improved over *odds ratio, information gain, Chi-Squared, BNS+F1* and *tficf* on unbalanced datasets like OHSUMED and Genomics, where majority of documents belong to only 2-3 categories. Our analysis showed that selecting the features that point to only one category performs better than selecting features that point to more than one category. Words that point to more than one category may mislead the classifier and hence decrease the effectiveness of a classifier on unbalanced datasets.

Furthermore, we provided analysis of how the micro-F1 is affected as we set more stringent thresholds for feature selection. We demonstrated that as the threshold for selecting the features is increased, the micro-F1 measure improves until up to a specific threshold. The training time for applying AM on Naïve Bayes classifier is not affected by feature selection algorithm. However, the time taken for training by SVM classifier is much lower than when no feature selection is used. The effectiveness of the text classifier decreases as the threshold increases beyond a certain point.

ACKNOWLEDGEMENTS

We like to thank Jay Urbain for providing us the preprocessed and indexed Genomics dataset.

REFERENCES

Breiman, L., Friedman, J., & Olshen, R. (1984). *Classification and regression trees.* Wadsworth International Group.

Chih, H., & Kulathuramaiyer, N. (2004). An Empirical Study of Feature Selection for Text Categorization based on Term Weightage. *IEEE/WIC/ACM International Conference on Web Intelligence*, (pp. 599-602).

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning , 20* (3), (pp. 273-297).

Debole, F., & Sebastiani, F. (2003). Supervised term weighting for automated text categorization. *ACM symposium on Applied computing*, (pp. 784-788).

Forman, G. (2008). BNS Scaling: An Improved Representation over TF·IDF for SVM Text Classification. *ACM Conference on Information and Knowledge Management,* (pp. 263-270).

Forman, G. (2004). A pitfall and solution in multi-class feature selection for text classification. *Proceedings of the 21st International Conference on Machine Learning*, (pp. 38-46).

Forman, G. (2003). An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research , 3*, (pp. 1289-1305).

Galavotti, L., & Sebastiani, F. (2000). Experiments on the Use of Feature Selection and Negative Evidence in Automated Text Categorization. *4th European Conference on Research and Advanced Technology for Digital Libraries.*, (pp. 59-68).

Grossman, D., & Frieder, O. (2004). *Information Retrieval: Algorithms and Heuristics.* 2nd Edition, Springer Publishers.

Hersh, W., Buckley, C., Leone, T., & Hickman, D. (1994). OHSUMED: an interactive retrieval evaluation and new large text collection for research. *Annual ACM Conference on Research and Development in Information Retrieval*, (pp. 192 - 201).

Joachims, T. (1999). Making Large-scale support vector machine learning practical. In B. S. al., *Advances in kernel methods: Support vector learning* (pp. 169–184).

Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with many relevant features. *10th European Conference on Machine Learning*, (pp. 137-142).

Lavelli, A., Sebastiani, F., & Zanoli, R. (2004). Distributional term representations: an experimental comparison. *thirteenth ACM international conference on Information and knowledge management*, (pp. 615 - 624).

Lewis, D. (1991). Evaluating Text categorization. *In Proceedings of the workshop on Speech and Natural Language*, (pp. 312-318).

Mccallum, A., & Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. *AAAI/ICML-98 Workshop on Learning for Text Categorization*, (pp. 41-48).

Mengle, S., & Goharian, N. (2008). *Using Ambiguity Measure Feature Selection Algorithm for Support Vector Machine Classifier.* ACM 23rd Symposium on Applied Computing, (pp. 916-920)

Mengle, S., Goharian, N., & Platt, A. (2007). FACT: Fast Algorithm for Categorizing Text. *IEEE 5th International Conference on Intelligence and Security Informatics*, (pp. 308-315).

Michie, D., Spiegelhalter, D., & Taylor, C. (1994). *Machine Learning, Neural and Statistical Classification.* Prentice Hall.

Mladenić, D., & Grobelnik, M. (1998). *Feature selection for classification based on text hierarchy.* Text and the Web, Conference on Automated Learning and Discovery CONALD-98.

Mladenić, D., Brank, J., Grobelnik, M., & Milic-Frayling, N. (2004). Feature Selection using Linear Classifier Weights: Interaction with Classification Models. *In The 27th ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 234-241).

Novovicova, J., & Malik, A. (2005). Information-theoretic feature selection algorithms for text classification. *IEEE International Joint Conference on Neural Networks*, (pp. 3272- 3277).

Porter, M. (1997). An algorithm for suffix stripping. *Morgan Kaufmann Multimedia Information And Systems Series*, (pp. 313 - 316).

Quinlan J. (1986) Induction of decision trees. Machine learning 1(1). (pp. 81-108)

Rennie, J., Teevan, J., & Karger, D. (2003). Tackling the poor assumptions of Naive Bayes Text classifiers. *20th International Conference on Machine Learning*, (pp. 616-623).

Schwartz, A., & Hearst, M. (2003). A simple algorithm for identifying abbreviation definitions in biomedical text. *Pacific Symposium on Biocomputing*, (pp. 451-462).

Urbain, J., Goharian, N., & Frieder, O. (2007). Combining Semantics, Context, and Statistical Evidence in Genomics Literature Search. *IEEE Seventh Symposium on Bio-Informatics & Bio-Engineering*, (pp. 1313 - 1317).

Wenqian, S., Houkuan, H., Haibin, Z., & Yongmin, L. (2007). A novel feature selection algorithm for text classification. *Expert Systems with Applications: An International Journal , 33* (1), (pp.1-5).

Yan, J., Liu, N., Zhang, B., Yan, S., Chen, Z., & Cheng, Q. (2005). OCFS: optimal orthogonal centroid feature selection for text categorization. *28th annual international ACM SIGIR conference on Research and development in Information Retrieval*, (pp. 122-129).

Yang, Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval , 1* (1-2), 69 - 90.

Yang, Y., & Pedersen, J. (1997). A comparative study on feature set selection in text categorization. *14th International Conference on Machine Learning*, (pp. 412-420).

Yang, Y., Zhang, J., & Kisiel, B. (2003). A scalability analysis of classifiers in text categorization. *26th ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 96-103).

Yi, L., & Zheng, Y. (2005). One-against-all multi-class SVM classification using reliability measures. *Conference on Neural Networks*, (pp. 849- 854).