Feature Selection Methods for Text Classification

Nicolette Nicolosi

November 7, 2008

Abstract

Text classification is an important and well studied area of pattern recognition, with a variety of modern applications. Effective spam email filtering systems, automated document organization and management, and improved information retrieval systems all benefit from techniques within this field. The problem of feature selection, or choosing the most relevant features out of what can be an incredibly large set of data, is particularly important for accurate text classification. This paper provides a brief overview of the area of text classification, followed by a survey of several popular feature selection methods commonly used for text classification. Pruning techniques are also briefly discussed as a way to further reduce the set of possible features (typically words) within a document prior to applying a method of feature selection.

1 Introduction

The purpose of text classification is to use the contents of a text or document to assign it to one or more categories. It has applications in document organization and management, information retrieval, and certain machine learning algorithms described in [5]. More effective spam email filtering systems, improved web search results, and better translations between languages can result from improved text classification techniques.

Feature selection forms an important subset within the much larger area of text classification. Correctly identifying the relevant features in a text is of vital importance to the task of text classification. Additionally, other methods for reducing dimensionality, such as pruning and clustering, can improve performance of text classification. This paper first briefly describes the current state of the area of text classification, and then examines different reasons and methods for reducing dimensionality, followed by common methods for identifying the most relevant features in a text. Some form of text classification generally follows the identification of the relevant features.

1.1 Background

Text classification is an area within pattern recognition and classification that has been studied with increasing frequency as Internet usage becomes more commonplace. The goal
is generally to assign a text to one or more classes based on some method that takes into 
account the contents of the text. There are many varied practical applications of text 
classification. The most well known of these applications is likely improved spam filtering 
techniques. Search engines may also take advantage of text classification techniques to return 
more accurate results to the user [5].

Common general techniques for text classification include both unsupervised and super-
vised pattern classification methods. Some common approaches use clustering instead of 
simple feature selection [2, 6, 10, 14], linear discriminant methods [1, 3], neural networks
[9], and support vector machines [17]. Some models attempt to use linguistic information
in the classification process, such as labeled samples from the WordNet data set [1, 11].
and empirical study of 12 feature selection metrics. With the exception of clustering, these
classification techniques are briefly discussed in the following section. Clustering is discussed
in greater detail with other feature selection techniques that are the focus of this paper.

1.2 Common Approaches

1.2.1 Linear Discriminant Models

The Rocchio classifier uses the concept of centroids to define decision boundaries. In terms
of Euclidean distance, a centroid of a class is computed by finding the vector average (center
of mass) of all members of the class. The boundary, then, is a set of points that has an equal
distance from two centroids. The generalized classifier can distinguish multiple classes. The
Rocchio relevance feedback algorithm calculates the centroid using a similarity metric to find
the most important part of the vector; however, it can only differentiate between relevant
and irrelevant [1].

1.2.2 Neural Networks

Neural networks have been used to solve a variety of pattern recognition problems. [9]
presents a neural network based approach to text classification that uses a Naive Bayes clas-
sifier. The initial state of the neural network is constructed using domain knowledge rather
than random weights. These initial weights are learned using the Naive Bayes classifier, and
the neural network generalizes this domain knowledge to attempt to produce an improved
text classification over the Naive Bayes classifier or neural network alone.

1.2.3 Linguistic Approaches

Linguistic approaches to text classification attempt to improve performance by taking into
account semantic information derived from natural language processing techniques.

[1] combines symbolic modeling typically used in NLP approaches with quantitative tech-
niques, such as the Rocchio classifier.

One claim is that the words contained within the training set are not enough by them-
selves to result in a universal model for a category. [11] discusses using the WordNet lexical
database in addition to the training set words to produce a more accurate model that is capable of taking into account word relationships, unlike a simple vector representation. The authors use WordNet features, part of speech tags, and term weighting schemes with a text classification model that uses a Naive Bayes classifier and a Support Vector Machine to differentiate between two categories. The authors found that incorporating the extra data did not have a noticeable affect on the accuracy of the tested classifiers; however, this approach may still be valid under different circumstances that have not yet been studied. The authors discuss several possible reasons for their results and offer several areas that may result in future improvements.

1.2.4 Combined Classifiers

There are also a number of different approaches that have attempted to combine classifiers to improve performance. Typically, a number of classifiers will be used and a voting system will be used to determine which classification is correct. [9] combines a Naive Bayes classifier and a neural network, but uses one to produce input for the other, rather than using several independent classifiers with voting.

[4] combines three classifiers to attempt to improve performance over single text classifiers. The first classifier is based on Rocchio’s Algorithm for relevance feedback, the second is a Naive Bayes classifier, and the third is a k-Nearest Neighbor classifier that uses similarity based on the difference between the test document and each neighbor in the training set. Rocchio’s Algorithm considers documents as vectors in vector space so that similar documents have similar vectors. The combined classifier was tested using both a simple majority voting scheme and a weighted majority voting scheme. A combined feature selector was also used. This selector pruned words that were at extremes in frequency (most and least frequent) and then chose words that were ranked according to amount of mutual information. The authors achieved 86% accuracy with the Naive Bayes classifier alone, 86.71% accuracy with the k-Nearest Neighbor classifier alone, and 87.57% accuracy with the Rocchio classifier alone. Using the first voting system, they achieved 88.29% accuracy. The second voting system achieved 88.14% accuracy, and the third 89.29% accuracy. The results of all methods tested by the authors were approximately comparable and while the combined classifier shows a slight increase in accuracy, it may be that differences in accuracy would only manifest on smaller or larger data sets than the 2000 documents used in the experiment.

1.3 Challenges and Limitations

One of the main challenges in performing text classification tasks is the high dimensionality of these types of problems. [15] A typical text document will contain hundreds of possible features, and it can be extremely difficult to produce an accurate classification without some sort of dimensionality reduction and feature selection and ranking techniques. Choosing the correct technique or techniques to use without losing important data can be difficult, and frequently requires experimentation with a variety of different methods. Additionally,
dimensionality reduction techniques can reduce over-fitting by the classifier to the training data. [15]

Text classification techniques can also frequently be very closely related to the AI-hard problem of natural language processing. It is then tempting to use some sort of natural language-based metrics in identifying features and producing classifications. However, attempts to use linguistic approaches have generally been unsuccessful so far. While [11] found that the addition of linguistic data, such as word relationships, did not significantly improve performance of the tested classifiers, they also reported that it did not have a significant negative affect on performance.

The performance of various text classification techniques is discussed in [15], but this paper does not contain the most recent results of current work in the field. It is, however, an excellent comparison of different types of techniques. Typical accuracy rates were reported to be around 80% to 85% on the same data set (Reuters). Some systems performed abysmally (15%), while some performed exceptionally well (92%).

2 Feature Selection

2.1 Purpose

The purpose of feature selection is to determine which features are the most relevant to the current classification task. In text classification, features are typically words from a document. Choosing an appropriate feature selection method for text classification can be vital because of the large number of features usually present in text documents.

Accurate feature selection is central to the performance of any of the methods discussed in the previous section, and there has been recent research concerning improved identification of the relevant features [5, 7, 12, 13, 18]. [5] presents an unsupervised feature selection algorithm that uses a randomly selected sampling-based strategy. [7] is an in-depth performance comparison of twelve feature selection methods evaluated on the same set of 229 classification problems, with the aim of improving overall performance rather than reducing the number of features required for acceptable performance. [12] briefly summarizes a number of experiments using combined methods for feature selection to attempt to improve overall precision in choosing the correct features. [13] discusses the results of a comparative study of several variants of five common feature selection methods. [18] takes a different approach in proposing the idea of partitioning substrings into equivalence groups followed by using a suffix tree algorithm to identify important groups within the text.

2.2 Common Approaches and Techniques

2.2.1 Pruning

Pruning methods are typically applied prior to feature selection to reduce the number of possible features. They are particularly important because the number of possible features is typically very large in a text document, and it is likely that most of these features are
irrelevant. Generally, very rare and very frequent words are commonly eliminated. For example, any word appearing two or fewer times is most likely irrelevant. Similarly, the most frequent words are also unlikely to be relevant - for example, "the," "a," "and," "or," etc... are very common in English, but are not usually the features that are useful for classifying a text document. [7] posits that word frequencies generally follow a Zipf distribution, which states that the frequency of a word is proportional to $\frac{1}{\text{rank}^p}$, where $\text{rank}$ is the place the word holds in a list of all words sorted by frequency, and $p$ is a fitting factor close to 1.0. The choice of threshold for when words are eliminated can have an effect on accuracy, so it is often vital that this chosen properly. Eliminating extremely frequent words can be done using a threshold or a list of stopwords, which are language and domain specific.

2.2.2 Random

[7] briefly describes a random feature selection metric, which was designed for use as a control in the experiment performed by the authors. They cite a study that claims that it scored very high for precision, but had a very low recall rate. The feature selector randomly ranks all features. This appears to be a good method of establishing a baseline for any experimentation with different feature selection metrics.

2.2.3 Accuracy and Balanced Accuracy

One of the intuitively simplest metrics is the calculated expected accuracy for a simple classifier built to recognize a single feature. It can be found by taking the difference between the true positives (tp) and the false positives (fp) to determine how many times the correct feature is selected. The balanced accuracy takes the difference between the true positive rate (tpr) and the false positive rate (fpr) rather than the numbers of true and false positives. The main benefit of this is the removal of a strong preference for a low false positive rate (fpr). The true positive rate $tpr = \frac{tp}{(tp+fn)}$, where $fn$ is the number of false negatives. Similarly, the false positive rate $fpr = \frac{fp}{(fp+tn)}$, where $tn$ is the number of true negatives. [7]

$$Acc = tp - fp$$

$$Acc2 = |tpr - fpr|$$

[7] cautions that Acc2 will select several negative features.

2.2.4 Chi-Squared

Chi-Squared is a statistical feature selection method that is widely applied within the field of pattern recognition. It measures divergence from the expected distribution, assuming that feature occurrence is independent of class value. It also generalizes well to multi-class problems. It has been shown to not work as well on very small expected counts. [7] It is
based on the chi-squared distribution in the fields of probability and statistics. [16] defines the $\chi^2$ function as follows

$$\chi^2(x, y) = \frac{N(O_{upcp} \times O_{upcn} - O_{upcp} \times O_{upcn})^2}{O_{up} \times O_{up} * O_{cp} * O_{cn}}$$

where $N$ is the total number of training samples, $M$ is the number of positive samples, $x$ is the number of samples that contain a subsequence $u$, $y$ is the number of samples in the positive class that contain the subsequence $u$, $c_p$ is the positive class, and $c_n$ is the negative class.

### 2.2.5 Information Gain

Information Gain (IG) is a method that measures the decrease in entropy when the feature is given rather than not given. It can generalize to any number of classes [7]. The entropy of a discrete random variable $X$ with a probability distribution $p(x)$ is defined as

$$H(p) = -\sum_{x \in X} p(x) \log p(x)$$

[6] discussed two methods for calculating the divergence, or relative entropy, between two probability distributions. The first is the Kullback-Leibler (KL) divergence between probability distributions $p_1(x)$ and $p_2(x)$, defined by the aforementioned paper as

$$KL(p_1, p_2) = \sum_{x \in X} p_1(x) \log \frac{p_1(x)}{p_2(x)}$$

The second method discussed is the Jensen-Shannon (JS) divergence between probability distributions $p_1(x)$ and $p_2(x)$, which is defined as follows for $p_1, p_2$, and $\pi_1 + \pi_2 = 1, \pi \geq 0$:

$$JS_\pi(p_1, p_2) = \pi_1 KL(p_1, \pi_1 p_1 + \pi_2 p_2) + \pi_2 KL(p_2, \pi_1 p_1 + \pi_2 p_2)$$

KL-divergence determines the distance between two probability distributions, but has some undesirable properties. While it is always positive, it can be unbounded. For example, $KL(p_1, p_2) = \infty$ when $p_1(x) = 0$ and $p_2(x) = 0$. It is also not symmetric, so it is not considered a true metric [6]. Jensen-Shannon divergence is symmetric and bounded, and can be generalized for more than two probability distributions.

The information gain function referenced by [6, 5] is derived from the entropy. In terms of the entropy equation $H(p)$, the information gain $I(X; Y)$ represents the reduction in entropy that occurs from one variable having knowledge of the other. Written in terms of entropy

$$I(X; Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

We can substitute the entropy equation to produce the information gain equation

$$I(X; Y) = \sum_x \sum_y p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$
[5] features a slight variation that measures the entropy decrease based on comparison with the entire set of features rather than two arbitrary features. This variation is more useful to practical feature selection for text, as opposed to the more theoretical definition given above. The equation for the variation is as follows

\[ IG(t) = \sum_{c \in \{c_k, \neg c_k\}} \sum_{t \in \{t_i, \neg t_i\}} P(t, c) \log \frac{p(t,c)}{p(t)p(c)} \] (9)

This metric can be applied by selecting the features that result in the largest decrease in entropy when they are removed from the set of all possible features. Information gain is a supervised feature selection method [5]. Information gain is also called expected mutual information. The mutual information function for a single instance [15] is

\[ MI(t_k, c_i) = \log \frac{P(t_k, c_i)}{P(t_k) * P(c_i)} \] (10)

[15] makes the claim that dimensionality can be reduced by a factor of 100 by information gain techniques with no negative impact on performance.

### 2.2.6 Probability Ratio

[7] defines the equation for the probability ratio as follows

\[ PR = \frac{\text{sample true positive rate}}{\text{sample false positive rate}} \] (11)

The sample true positive rate is defined as the number of true positives divided by the number of positive cases. Similarly, the false positive rate is defined as the number of false positives divided by the number of negative cases. This metric produces the same decision surface as the log probability ratio (\( \log(tpr/fpr) \)), but is faster to compute [7]. The probability ratio intuitively represents the percentage of times there is an actual positive or negative for the given sample. The probability ratio can be used to produce a ranking of features based on how frequently they are correctly identified as being relevant.

### 2.2.7 Document Frequency

The document frequency metric is an unsupervised feature selection method that simply counts the number of documents in which the given feature appears [5]. This can be computed without class labels, so it can be safely found for the test set. [7] However, it appears that this is generally performed on the training set to maintain separation between the training and test sets. The equation for document frequency given by [7] is

\[ Dfreq = \text{number of true positives} + \text{number of false positives} \] (12)

[15] claims that it is possible to use document frequency to reduce dimensionality by a factor of 10 with no loss of classifier performance, and by a factor of 100 with only a small negative effect on performance.
2.2.8 Bi-Normal Separation

[7] describes the Bi-Normal Separation feature selection metric. It takes the difference between the values for the standard normal distributions inverse cumulative probability function ($F^{-1}$) on tpr (sample true positive rate) and fpr (sample false positive rate). The equation is given as

$$BNS = F^{-1}(tpr) - F^{-1}(fpr)$$

This represents the occurrence of a feature in a document as an event of a random normal variable exceeding some threshold. The area under the curve past this threshold is the prevalence rate. This metric measures the separation between two thresholds.

2.2.9 Sampling-based Techniques

The focus of [5] is sampling-based feature selection strategies. The authors attempt to create an unsupervised feature selection metric that chooses a small number of features that preserve the geometric structure of the data. The next three methods discussed are described in this paper and are all based on this general concept. An $n$x$n$ diagonal sampling matrix $S$ is defined, and the diagonal entries are determined randomly by ”coin flips.” An $n$x$d$ feature-document ranking matrix $A$ is also defined. The $d$ columns in $A$ correspond to $d$ objects in an $n$-dimensional feature space.

2.2.10 Subspace Sampling

Subspace sampling defines the probability $p_i$ of choosing each feature as

$$p_i = \frac{||U_{k(i)}||^2_2}{2}, \quad (14)$$

where $U_k$ is a matrix representing the top $k$ left singular vectors of $A$, and $||U_{k(i)}||^2_2$ is the squared length of the row corresponding with $k$. [5]

2.2.11 Weight-based Sampling

Weight-based Sampling defines the probability $p_i$ of choosing each feature as

$$p_i = ||A(i)||^2_F||A||^2_F, \quad (15)$$

or $p_i$ is proportional to the squared length of the corresponding row in the matrix $A$. [5]

2.2.12 Uniform Sampling

Uniform Sampling defines the probability $p_i$ of choosing each feature as equal to all other features. [5] So

$$p_i = \frac{1}{n}, \text{ for } i = 1, \ldots, n \quad (16)$$
2.2.13 Clustering

A large number of approaches take advantage of clustering in building text classifiers. This technique attempts to reduce the dimensionality of text data by treating clusters of related word features as a single feature. Word clustering can increase classification accuracy for problems in which there exist small training sets or noisy feature sets [6]. Clustering differs from selection in that clustering attempts to group synonymous terms, while selection attempts to choose only informative terms. [15]

[6] attempts to improve on existing clustering techniques that are computationally expensive and frequently produce suboptimal word clusters. The authors present a divisive algorithm for feature clustering that achieves a higher cluster accuracy on a low number of features. The clustering algorithm attempts to minimize the value of a global criterion function, and is similar to the $k$-means algorithm. The chief difference is that Kullback-Leibler divergences are used rather than squared Euclidean distances. The KL divergence measures the ”distance” between probability distributions, and is always non-negative. It is also known as the relative entropy between two probability distributions. Again, it is defined as follows for probability distributions $p_1(x)$ and $p_2(x)$:

$$KL(p_1, p_2) = \sum_{x \in X} p_1(x) \log \frac{p_1(x)}{p_2(x)}$$ (17)

This feature clustering technique was used to build smaller class models for hierarchical classification. The classifier built using this model used Support Vector Machines and Naive Bayes to classify data from the 20-Newsgroups data set and a hierarchy of HTML documents from the Dmoz Open Directory with a depth of 3 levels. This method was found to achieve a higher classification accuracy on sparse data.

2.3 Performance

There have been many studies comparing performance on feature selection methods for text classification. [7] compared performance of 12 feature selectors and reported that Chi-Squared and information gain performed best on multi-class problems. They tested these selectors on 19 multi-class data sets that contained 229 binary text classification problems. Interestingly, they found that a Support Vector Machine using all available features performed particularly well, and that using Bi-Normal Separation can typically improve upon this performance. Information Gain was also found to perform well. The authors also varied the threshold for removal of rare words and found that it is typically best to set the threshold low and then use feature selection to avoid a negative effect on classification.

However, [8] claims that methods such as Chi-Squared and information gain that perform independent feature scoring are susceptible to being distracted by strongly predictive features for easier classes, thereby not selecting useful features for the difficult classes. The authors suggest using feature allocation policies to attempt to distribute a fair amount of attention to each class. They suggest that this could be done based on class distribution or some other estimated cost weighting method.
3 Conclusion

Accurate feature selection techniques are generally vital to the performance of text classification systems. This is because of the typically vast size of sets of potential features in these problems. In a text document, each word can be a possible feature. These large numbers of possible features can typically be reduced through a variety of techniques. Pruning, feature selection, and clustering are methods for reducing the dimensionality of a test classification problem. Pruning removes extremely rare and extremely common words from the set of possible features, typically prior to the application of any feature selection metrics. Feature selection determines the most relevant features, or words, in a document. Clustering is a method by which related features are considered together as a single feature. This paper has provided a brief introduction to some current work in the area of text classification, followed by definitions and examples of a number of common methods for pruning, feature selection, and clustering. It is possible to combine these methods to further reduce dimensionality, but one must be careful to avoid removing too many possible features as this can adversely affect performance.

References


