Feature Preparation in Text Categorization

Ciya Liao, Shamim Alpha, Paul Dixon
Oracle Corporation,
{david.liao, shamim.alpha, paul.dixon}@oracle.com

Abstract

Text categorization is an important application of machine learning to the field of document information retrieval. Most machine learning methods treat text documents as a feature vectors. We report text categorization accuracy for different types of features and different types of feature weights. The comparison of these classifiers shows that stemmed or un-stemmed single words as features give better classifier performance compared with other types of features, and LOG(tf)IDF weight as feature weight gives better classifier performance than other types of feature weights.

Introduction

Text categorization is a conventional classification problem applied to the textual domain. It solves the problem of assigning text content to predefined categories. As the volume of text content grows continuously on-line and in corporate domains, text categorization, acting as a way to organize the text content, becomes interesting not only from an academic but also from an industrial point of view. A growing number of statistical classification methods have been applied to text categorization, such as Naive Bayesian [Joachims,97], Bayesian Network [Sahami,96], Decision Tree [Quinlan,93] [Weiss,99], Neural Network[Wiener,95], Linear Regression[Yang,92], k-NN [Yang,99], Support Vector Machines [Dumais,98][Joachims, 98], and Boosting [Schapire,00][Weiss,99]. A comprehensive comparative evaluation of a wide-range of text categorization methods is reported in ref.[Yang,99][Dumais,98] against the Reuters corpus.

Most of the statistical classification methods mentioned above are borrowed from the field of machine learning, where a classified item is treated as a feature vector. A simple way to transform a text document into a feature vector is using a "bag-of-words" representation, where each feature is a single token. There are two problems associated with this representation.

The first problem to be raised when using a feature vector representation is to answer the question, "what is a feature?". In general, a feature can be either local or global. In text categorization, local features are always used but in different-length scales of locality. A feature can be as simple as a single token, or a linguistic phrase, or a much more complicated syntax template. A feature can be a characteristic quantity at different linguistic levels. To transform a document, which can be regarded as a string of tokens, into another set of tokens will lose some linguistic information such as word sequence. Word sequence is crucial for a human being to understand a document and should be also crucial for a computer. Using phrases as features is a partial solution for incorporating word sequence information into text categorization. This paper will investigate the effectiveness of different classifiers by using single tokens, phrases, stemmed tokens, etc. as features.

The second problem is how to quantify a feature. A feature weight should show the degree of information represented by local feature occurrences in a document, at a minimum. A slightly more

complicated feature weight scheme may also represent statistical information of the feature's occurrence within the whole training set or in a pre-existing knowledge base (taxonomy or ontology). A yet more complicated feature weight may also include information about feature distribution among different classes. This paper will only investigate the first two types of feature weights.

From Text to Features

In order to transform a document into a feature vector, preprocessing is needed. This includes feature formation (tokenization, phrase formation, or higher level feature extraction), feature selection, and feature score calculations. Tokenization is a trivial problem for white-spaced languages like English.

Feature formation must be performed with reference to the definition of the features. Different linguistic components of a document can form different types of features. Features such as single tokens or single stemmed tokens are most frequently used in text categorization. In this bag-of-words representation, information about dependencies and the relative positions of different tokens are not used. Phrasal features consisting of more than one token are one possible way to make use of the dependencies and relative positions of component tokens. Previous experiments [Sahami,96] [Dumais,98] show that introducing some degree of term dependence in the Bayesian network method will achieve undoubtably higher accuracy in text categorization compared to the independence assumption in the Naive Bayesian method. However, whether the introduction of phrases will improve the accuracy of text categorization has been debated for a long time. Lewis [Lewis,92] was the first to study the effects of syntactic phrases in text categorization. In his study, a naive Bayesian classifier with only noun phrases yielded significantly lower effectiveness than a standard classifier using bag-of-single-words. More reports on inclusion of syntactic phrases show no significant improvement on rule-based classifiers [Scott,99] and naive Bayesian and SVM classifiers [Dumais,98]. For statistical phrases like n-grams, one report [Caropreso,01] shows that certain term selection methods such as document frequency, information gain and chi-square give high selection scores to a considerable number of statistical phrases, which indicates they have important predictive value. In the same report, directly using selected uni-grams or bigrams during text categorization with the Rocchio classifier yields a slightly higher effectiveness compared to only using uni-grams in the case that the classifier chooses an adequate but equal number of terms as features. A significant drop in effectiveness was observed when the classifier chose fewer terms. The report then commented that inclusion of some bigrams may only duplicate information of existing uni-grams but force other important uni-grams out. However, other reports on statistical phrases show that the addition of n-grams to the single words model can improve performance in the shorter-length n-grams case [Furnkranz, 98] [Mladenic,98].

One type of a higher level feature has been studied in text categorization [Riloff,98], where linguistic patterns were extracted automatically and input as features to naive Bayesian and rule-based classifiers. A consistent improvement in precision was observed in the naive Bayesian classifier and at low recall level in the rule-based classifier. Adding linguistic patterns to the single word representation yields consistent improvement of precision except at a very high recall level.

Feature selection has been studied by [Yang,97], where information gain and chi-square methods are found most effective for k-NN and linear regression learning methods. Term selection based on document frequency in the training set as a whole is simple but has similar performance to information gain and chi-square methods.

Selected features must be associated with a numerical value to evaluate the

impact of the feature to the classification problem. Most types of feature weighting schemes in text categorization are borrowed from the field of information retrieval. The most frequently used weight is TFIDF [Salton, 1988]. The original TFIDF is:

$$\omega_{fd} = tf_{fd} \log \frac{D}{df_f} \qquad \text{eq. 1}$$

where ω_{fd} is the weight of feature f in document d, tf_{fd} the occurrence frequency of feature f in document d, D the total number of documents in the training set, and df_f is the number of documents containing the feature f.

In this paper, we will compare text categorization using different types of features, and different types of feature weighting schemes. The feature types will include single tokens, single stemmed tokens, and phrases. Weighting schemes will include binary feature (BI), term frequency (TF), TFIDF(eq.1), logTFIDF(eq.2), etc.

$$\omega_{fd} = \log(tf_{fd} + 0.5) \log \frac{D}{df_f} \qquad \text{eq. 2}$$

We note that the logarithm of the TF part is to amend unfavorable linearity. The machine learning algorithms we report in this paper include SVM [Joachims, 98] and Neural Network. Feature selection in Neural Networks and Support Vector Machine classifiers is based on document frequency. Only features (single words or phrases) occurring in an adequate number of training documents will be selected. The corpus includes reuters-21578 and ohsumed.

Phrase Features

We only use training set documents to find valid phrases. We first scan the documents in the training set and detect phrases based on linguistic and statistical conditions. We only use noun phrases as valid phrases. Valid phrases are inserted into a phrase database which is specific to the training set.

The phrase database is used to replace the phrases in the training documents and test documents with specific tokens. For example, the phrase "information retrieval" in the document will be changed to the token "information_retrieval". After phrases in the documents are marked, the documents can be input into tokenization program in training or classification processes for performance testing.

To detect valid noun phrase chunking, Brill's transformation-based part of speech tagger [Brill, 1995] was used to mark parts-of-speech in the training documents. Training documents with POS tags are input into Ramshaw&Marcus's noun phrase chunking detector [Ramshaw,95] for noun phrase detection. The resultant noun phrase chunks are output to a file, which is input to a statistical chi-square test program. This program tests the statistical significance of co-occurrences of the component tokens in n-gram noun phrases. In particular, we choose the noun phrases (ngrams, up to 4-grams) such that the null hypothesis that its component tokens are independent of each other can be proved not true.

Machine Learning Algorithms

We test two different type of machine learning algorithms: Neural Networks and Support Vector Machines. We use the SVM_light package [Joachims, 98] with default parameter settings, which results in a linear SVM classifier.

For Neural Network, we use a home-made program. The Neural Network has no hidden layer and therefore is equivalent to a linear classifier. Text document classification has high dimensional data characteristics because of the large size of natural language vocabulary. Documents in one class usually can be linearly separated from other classes due to high dimensionality [Joachims, 98] [Schutze, 1995]. A prior experiment [Schutze, 1995] shows that linear neural networks can achieve the same accuracy as non-linear neural networks with hidden layers.

During the learning process, a sequential back propagation algorithm is used to minimize training error. We use cross-entropy error, thus making our learning method equivalent to logistic regression learning [Schutze, 1995]. We tried to use weight regularization methods [Zhang, 2001] to deal with overfitting, but the accuracy was not improved and convergence is hard to achieve by using back propagation learning. The results we present in this paper do not use regularization.

Corpus

The evaluation experiments are done on two text collections. The first is Reuters-21578 with ModApte split. Many text categorization methods have been tested against this corpus [Yang,99] [Dumais,98] [Joachims,98]. This is a collection of newswire stories from 1987 compiled by David Lewis. The number of distinct tokens in the training set is 39189, of which 18586 tokens occur more than once, 12951 tokens occur more than twice, 10328 tokens occur more than three times, 8789 tokens occur more than four times, and 3262 tokens occur more than 20 times in the training set.

The second collection is taken from Ohsumed corpus used in the Filtering Track in TREC-9 [trec9 report]. The Ohsumed collection consists of Medline documents from the years 1987-1991 and a set of topics and relevance judgments. In order to reduce the size of the problem, we chose MESH categories in which the number of Ohsumed documents in 1991 is larger than 300 (which results 98 categories). The training/testing split is across the document series number 91250000. Training documents have the document series number less than 91250000. This split results in 14655 training documents and 6698 test documents. The resultant training set and testing set have more homogenous distribution across different categories than the Reuters collection. The minimum (maximum) number of training documents in one category is 65 (465). The minimum (maximum) number of testing documents in one category is 29(214). In the training set, there are 52162 distinct tokens, of which 28857 tokens occur more than once, 22128 tokens occur more than twice, 18493 tokens occur more than three times, 9224 tokens occur more than 12 times, and 3458 tokens occur more than 60 times.

Experiment Results and Discussion

We first compare the impact of different feature types. The meaning of the following legends in the figure denote feature types: "single words" means only single tokens as features, "noun phrases" uses detected noun phrases as features without using component tokens, "stem words" uses Porter-stemmer-determined stems for each token as features, "noun phrases and words" uses detected noun phrases and their component tokens.

Fig.1 and Fig.2 show the micro-average breakeven points (BEP) with different numbers of features using the SVM classifier to classify the Reuters and ohsumed corpora, respectively. Breakeven accuracies increase with the number of features. There is no overfitting observed in the experimental range as the number of features increases. The maximum number of features in Fig.1 is 16000 for Reuters. This number of single tokens is roughly the number of tokens occurring twice or above in the training set. The maximum number of features in Fig.2 is 20000 for ohsumed. This number of single tokens is roughly the number of tokens occurring three times or above in the training set.

The maximum BEPs are achieved at the maximum number of features. For reuters, the best BEP is 0.88, which is slightly higher than the reported microAvg. BEP in [Joachims, 98] (0.860). For ohsumed, the best BEP is 0.602, achieved by using stem words.

The effect of stemming can be easily seen in Fig. 1 and 2. When the number of features is small, the coverage of selected features is poor but stemming of words can increase the feature coverage, thus giving the best breakeven accuracy compared to other types of features. However, as the number of

chosen features increases, and coverage of the chosen features becomes large enough, the accuracy of the features in conveying the information becomes more important. This can be seen in Fig.1 where the BEPs of other types of features are as good as stem words at large number of features.

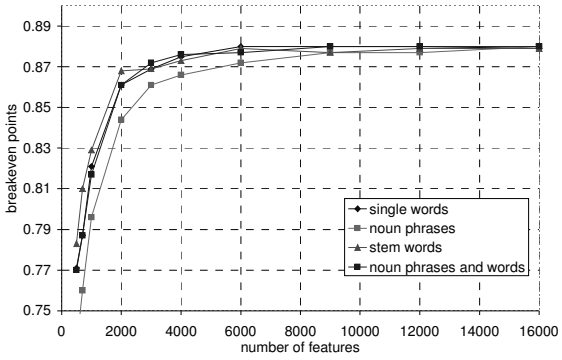

Fig.1 Breakeven points for the Reuters collection using SVM. The feature vector is normalized to have unity sum. The feature weight is LOGTFIDF.

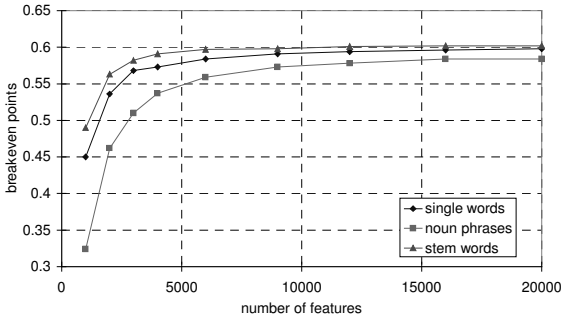

Fig.2 Breakeven points for the ohsumed collection using SVM. The feature vector is normalized to have unity sum. The feature weight is LOGTFIDF.

It is interesting to see how well the automatically determined noun phrase features perform. The Fig.1 and Fig. 2 show that noun phrases classifiers give the worst BEP. This is disappointing. It was expected that good phrase features provide more accurate information by constraining the meaning of component words. One example is the phrase "consumer price index", where the combination of the three token means a specific index. However, this result is consistent with the findings of the previous literature [Lewis,92], [Scott,99],[Dumais,98]. This can be partially explained by decreased feature coverage due to replacement of original tokens by phrases. For example, if the two words "oracle database" are replaced by one phrase "oracle_database", this phrase feature will only match itself and can not convey any similarity with its individual word components "oracle" and "database". However, we know that an article discussing enterprise software may only mention "oracle" or "database" separately. Another example is replacing the two phrases "Oracle database" and "DB2 database" with two separate features makes it impossible to map the similarity existing between them. Thus each phrase will have narrower coverage. The coverage problem is caused by replacing a number of token features with a single feature which is more accurate but finer. This problem can be partially reduced by not eliminating the phrase's component words. In fig.1, we see a significant increase of BEP for noun phrases including component words.

The single words are natural linguistic units and are employed by many text classification systems. From Fig.1 and Fig.2, one can see that this natural and simple feature unit performs fairly well compared with other complicated types of features.

The above results and discussion about types of features are not just applied to a single machine learning algorithm (SVM in Fig.1, Fig.2). We performed the same experiments with a Neural Network classifier. The results are shown in Fig.3 and Fig.4. The BEPs using Neural Network are not as good as those using SVM. The maximum BEP using Neural Network is 0.871 for Reuters using single words and 0.568 for ohsumed using stem words. It is worthwhile to mention that the training time for the neural network is much longer than SVM (>10 times in this experiment's problem scale).

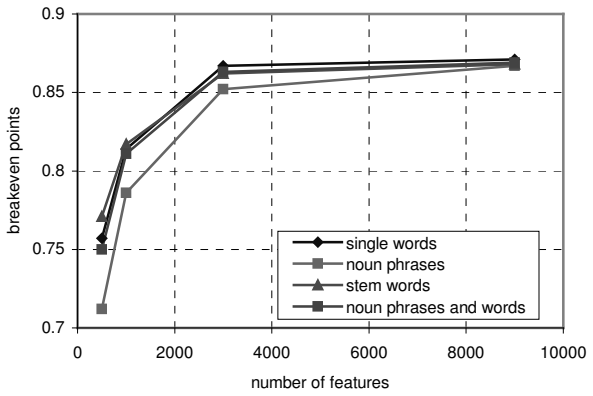

Fig.3 Breakeven points for the Reuters collection using Neural Network. The feature vector is normalized to have unity sum. The feature weight is LOGTFIDF

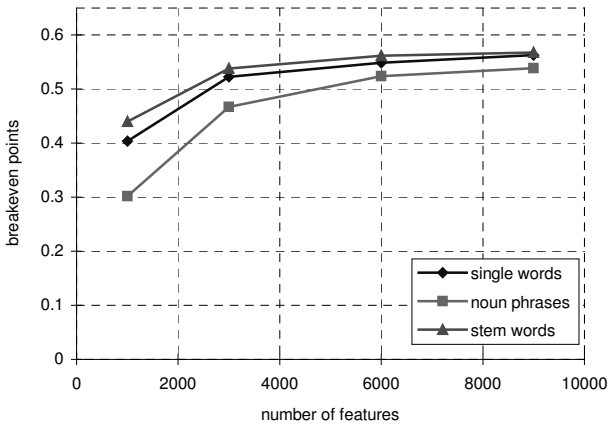

Fig.4 Breakeven points for the ohsumed collection using Neural Network. The feature vector is normalized to have unity sum. The feature weight is LOGTFIDF

So far, the discussed results all use one feature weighting scheme: LOGTFIDF (eq. 2). In Fig.5, Fig.6, we employed different feature weighting schemes. They are:

IDF: $\log(D/df_f)$
TF: tf_{fd}
TFIDF: eq. 1
LOGTFIDF: eq. 2
LOGTF: $\log(tf_{fd})$
BINARY: 1 or 0

Although the BEP ranking sequence for different weighting schemes is different for Fig.5 and Fig.6, we still can find some common characteristics for weighting methods based on the results vs. the Reuters and ohsumed collections. One can observe that the BEPs using LOGTF weight are always larger than those using TF weight. This shows that non-linear weighting of term frequency is better than conventional linear weighting. We think that this observation also holds in the field of information retrieval for relevance ranking.

Fig.5 and Fig.6 show that IDF weighting is better than BINARY weighting. Because IDF weight is only assigned to a feature occurring in the concerned document, IDF weight is actually BINARY weight multiplying an IDF score which contains the statistical information of the feature inside the whole corpus. Considering the fact that TFIDF is better than TF, LOGTFIDF better than LOGTF, one can than conclude that introducing the corpus information helps improve the accuracy of text categorization.

It is seen from Fig.5 and Fig.6 that LOGTFIDF, which is the multiplication of the LOGTF and IDF weights, performs the best in both collections.

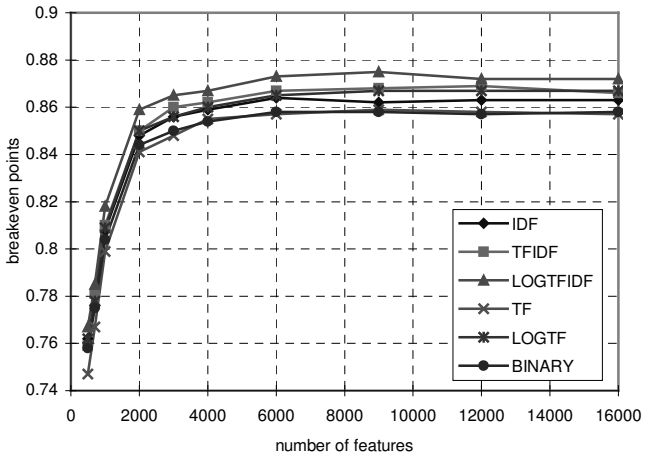

Fig.5 Breakeven points for the Reuters collection using SVM. The feature vector is normalized to have unity length. The features are single words.

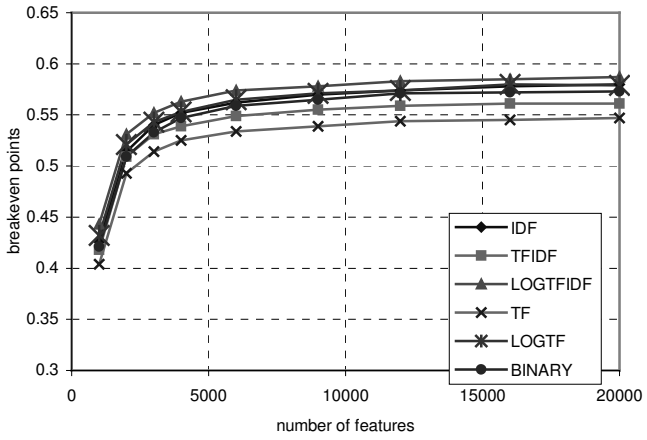

Fig.6 Breakeven points for the ohsumed collection using SVM. The feature vector is normalized to have unity length. The features are single words.

Conclusions

We have compared text classifiers using different types of features and different weighting schemes. We employed Support Vector Machines and a Neural Network algorithm as two linear classifiers. We tested these classifiers on the Reuters and ohsumed collections. Based on this comparison, we find that SVM algorithm is superior to the linearized neural network both in accuracy and training speed. Stem words, which normalize different feature forms to one stem form, show significant advantages in the case where a small number of features are used because of the larger coverage of the stems. Replacing contiguous tokens with detected noun phrases as features gains accuracy but loses coverage due to the problem of normalization. One may surmise that if a similarity match between different features is introduced to replace the current binary match between two features, the feature normalization problem will be eliminated. Single words as features perform fairly well. The comparison of different weighting schemes shows LOGTFIDF as the preferred feature weighting method.

References

[Joachims, 98] T.Joachims, Text categorization with support vector machines: learning with many relevant features. Proceedings of ECML-98, 10th European Conference on Machine Learning, 1998.

[Yang,97] Y.Yang, J.O. Pederson, A comparative study on feature selection in text categorization, International Conference on Machine Learning (ICML), 1997.

[Yang,99] Y.Yang, An evaluation of statistical approaches to text categorization. Journal of Information Retrieval, Vol.1, No.1/2, 1999.

[Weiss,99] S.M.Weiss,C.Apte,F.J.Damerau, D.E.Johnson,F.J.Oles,T.Goetz,T.Hampp, Maximizing text-mining performance,IEEE Intelligent systems,1999.

[Quinlan,93] J.R.Quinlan, C4.5: Programs for machine learning, Morgan Kaufmann, 1993.

[Joachims,97] T. Joachims, A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization, Proceedings of ICML-97, 14th International Conference on Machine Learning, 1997.

[Wiener,95] E. Wiener, J.O.Pederson, A.S.Weigend, A neural network approach to topic spotting. Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, 1995.

[Dumais,98] S.Dumais, J.Platt, D.Heckerman, M.Sahami, Inductive learning algorithms and representations for text categorization. Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM 98), 1998.

[Sahami, 96] M. Sahami,Learning limited dependence Bayesian classifier. In KDD-96: Proceedings of the second international Conference on Knowledge Discovery and Data Mining, 335-338. AAAI press, 1996.

[Yang, 92] Y.Yang, C.G.Chute, A linear least squares fit mapping method for information retrieval from natural language texts. Proceedings of the 14th International Conference on Computational Linguistics (COLING 92), 1992.

[Schapire,00] R.E.Schapire, Y.Singer, Boostexter: A boosting-based system for text categorization. Machine Learning, 39(2/3), 2000.

[Lewis,92] D.D.Lewis, Feature selection and feature extraction for text categorization, Proceedings of Speech and Natural Language Workshop, 1992.

[Scott,99] S.Scott, S.Matwin, Feature engineering for text classification, Proceedings of ICML-99, 16th International Conference on Machine Learning, 1999.

[Caropreso,01] M.F.Caropreso, S.Matwin, F.Sebastiani, A learner-independent evaluation of the usefulness of statistical phrases in automated text categorization, Text database and Document Management: Theory and Practice, 2001.

[Furnkranz, 98] J.Furnkranz, A study using n-gram features for text categorization. Technical Report OEFAITR-98-30. 1998.

[Mladenic,98] D.Mladenic, M.Grobelnik, Word sequences as features in text-learning. Proceedings of ERK-98, the Seventh Electrotechnical and Computer Science Conferences, 1998.

[Riloff,98] J.Furnkranz, T,Mitchell, E.Riloff, A case study of using linguistic phrases for text categorization on the WWW, Proceedings of the 1st AAAI Workshop on Learning for Text Categorization, 1998.

[Brill,95] Transformation-based error-driven learning and natural language processing: a case study in part of speech tagging, Computational Linguistics, 21(4),1995.

[Ramshaw, 95] L.A.Ramshaw, M.P.Marcus, Text chunking using transformation-based learning, Proceedings of third Workshop on Very Large Corpora, 1995.

[Salton, 1988] G. Salton, C. Buckley, Term Weighting Approaches in Automatic Text Retrieval, Information Processing and Management, Vol. 24, No.5, P513, 1988.

[Schutze, 1995] H. Schutze, D.A.Hull, J.O.Pederson, A Comprison of Classiifers and Document Representations for the Routing Problem, in Proceedings of SIGIR95, 1995.

[Zhang, 2001] T. Zhang, F.J. Oles, Text Categorization Based on Regularized Linear Classification Methods, Information Retrieval, vol.4, 2001.

[trec9 report] S. Robertson, D.A.Hull, The TREC-9 Filter Track Final report, Proceedings of the Ninth Text Retrieval Conference (TREC-9), 2000.