Support Vector Machines for Text Categorization

A. Basu, C. Watters, and M. Shepherd
Faculty of Computer Science
Dalhousie University
Halifax, Nova Scotia, Canada B3H 1W5
{basu | watters | shepherd@cs.dal.ca}

Abstract

Text categorization is the process of sorting text documents into one or more predefined categories or classes of similar documents. Differences in the results of such categorization arise from the feature set chosen to base the association of a given document with a given category. Advocates of text categorization recognize that the sorting of text documents into categories of like documents reduces the overhead required for fast retrieval of such documents and provides smaller domains in which the users may explore similar documents.

In this paper we are interested in examining whether automatic classification of news texts can be improved by a prefiltering the vocabulary to reduce the feature set used in the computations. First we compare artificial neural network and support vector machine algorithms for use as text classifiers of news items. Secondly, we identify a reduction in feature set that provides improved results.

1. Introduction

Text categorization is the process of sorting text documents into one or more predefined categories or classes of similar documents. Differences in the results of such categorization arise from the feature set chosen to base the association of a given document with a given category. Categorization may be based on human judgment, as is done by **Yahoo**, simple keyword clustering, or learning algorithms. Advocates of text categorization recognize that the sorting of text documents into categories of like documents reduces the overhead required for fast retrieval of such documents and provides smaller domains in which the users may explore similar documents.

Text categorization requires, as a basis, the identification of features within the documents that can be used to discriminate amongst the documents and associate individual documents to individual categories. These categories may be determined *a priori*, either by humans or algorithmically, or may be determined dynamically as needed. Information retrieval systems have used traditional classification schemes while most clustering algorithms use the vector space model [13] to form clusters of documents. The vector space model uses a sparse matrix of keyword occurrences which requires rebuilding for each new set of documents.

More recently, researchers have explored the use of machine learning techniques to automatically associate documents with categories by first using a training set to adapt the classifier to the feature set of the particular document set [7]. The machine learning process is initiated by an examination of sample documents to determine the minimal feature set that produces the expected categorization results. This training phase may be supervised or unsupervised. In both cases a set of categories has been defined *a priori*, unlike clustering which defines the categories based on features of the actual documents. The unsupervised learning techniques uses features of the training documents to let the algorithm determine the category each document belongs in. Supervised learning techniques use a set of training documents that have already been associated with a category to determine which feature set of the documents will produce the desired results. Machine learning techniques, if successful, provide an advantage in dynamic document sets, over the standard vector space model, in that the introduction of new documents and new document sets does not require rebuilding of the document vector matrices.

In this paper we compare an artificial neural net algorithm with a support vector machine algorithm for use as text classifiers of news items. We also identify a reduction in feature set that can be in both algorithms and test if this reduction affects the performance.

2. Background

2.1. Data Set

The Reuters News Data Sets are frequently used as benchmarks for classification algorithms. The Reuters-21578 collection [9] is a set of 21,578 short (average 200 words in length) news items, largely financially related, that have been preclassified manually into 118 categories. The mean number of classifications per document is 1.2, with some items not classified at all and some items assigned to 12 categories. The distribution of items in the categories is also uneven with the largest containing 3964 items and the smallest category with a single item. The ModApte split provides a division of these items into a training set of 9603 items and 3299 test items. These document sets are stored in SGML format and have been used as the document set in many experiments and trial systems.

2.2. Support Vector Machines (SVM)

SVM classification algorithms, proposed by Vapnik [14] to solve two-class problems, are based on finding a separation between hyperplanes defined by classes of data [1], shown in Figure 1. This means that the SVM algorithm can operate even in fairly large feature sets as the goal is to measure the margin of separation of the data rather than matches on features. The SVM is trained using preclassified documents.

Research has shown [8] that SVM scales well and has good performance on large data sets.

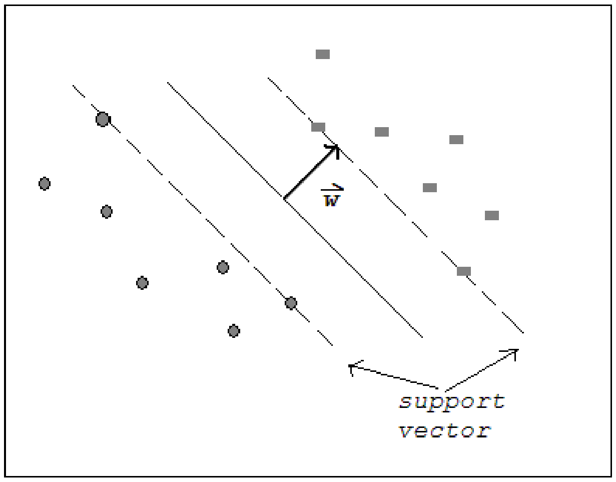


Figure 1. Example of SVM hyperplane pattern

Using the entire vocabulary as the feature set, Rennie and Rifkin [12] found that the SVM algorithm outperformed the Naïve Bayes algorithm used on two data sets; 19,997 news related documents in 20 categories and 9649 industry sector data documents in 105 categories. Naïve Bayes classification algorithms are based on an assumption that the terms used in documents are independent. Both Bayes and SVM algorithms are linear, efficient, and scalable to large document sets [12].

Joachims [7] used a reduced vocabulary as the feature set by first word stemming and then using a stop list of very frequent words and elimination of very infrequent words from the feature set. Using 12,902 documents from the Reuters-21578 document set and 20,000 medical abstracts from the Ohsumed corpus, Jaochims compared the performance of several algorithms including SVM, and Naïve Bayes. For both document sets, this test indicated that the SVM performed better.

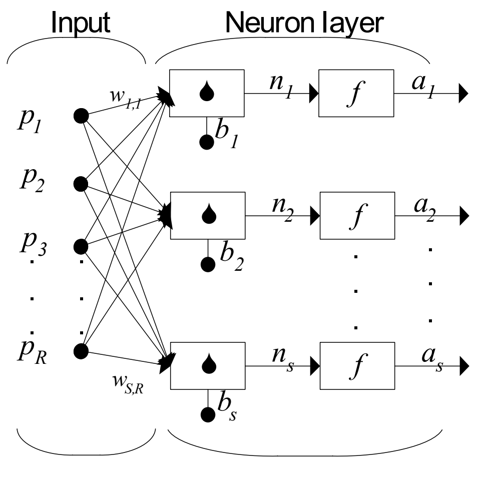
Dumais et al [6], using the Reuters-21578 collection, found that the SVM algorithm performed the most accurately in a test that compared the Naïve Bayes, Decision Trees, and SVM.

2.3. Artificial Neural Network

Artificial Neural Networks attempt to imitate the operation of natural neurons in the hope of realizing a similar function.

In the artificial neuron the movement of an impulse along the neuron is modeled by a scalar or vector value, and the alteration of the impulse is simulated using a transfer function. Therefore a simple artificial neuron can be modeled using the function, $a = f(wp + b)$ [5]. Where *p* is the input scalar, *w* is a scalar weight and *b* is the bias to move the function '*f*' in some direction. The transfer function *f* is typically a stepwise function (e.g., Hard Limit) or some sort of a sigmoid (e.g, Log-Sigmoid), but it can also be a linear function. This function takes a single parameter *n*, which is *wp*+*b*. If the input, *p*, is a vector *w* becomes a single row matrix. The function then becomes $a = f(wp + b)$, where the product is simply the dot product of *w* and *p*, and is shown in Figure 2.

Wemter et al [15] used a recurrent plausibility network for text categorization using the ModApte split from the Reuters-21578 news data set, which is a set of 10,733 documents belonging to one or more of only eight categories all tightly related to finance. The test results after training produce 93% recall and 92% precision on this data set.



$$\mathbf{a} = \mathbf{f}(\mathbf{wp} + \mathbf{b})$$

Figure 2. Artificial Neural Net

3. Methodology

3.1. Data Set

For this experiment we used the Reuters-21578 document set, which contains 21,578 documents in SGML format and 118 predefined categories. Most of the documents belonged in one or more categories but some documents were not allocated to any category and some were later determined to be mis-categorized.

The SGML documents were converted into XML format using the SX tool [3]. The documents were filed by category, and documents belonging to multiple categories were copied into each category. For our purposes, categories with fewer than fifteen documents, documents with no category assigned, and documents in the mis-categorized category were eliminated, leaving a set of 11,327 documents in 63 categories.

A global dictionary of terms was created using KSS [4]. This dictionary contained 102,283 distinct terms. Rather than use such a large feature set to define the document vector, techniques were used to provide a smaller more effective feature set of terms. The IQ value, produced by the KSS system, was used as a threshold to reduce the numbers of terms. The IQ measure, like the Inverse Document Frequency, measures the importance of a given term across the entire document collection in discriminating documents. In addition, the KSS system allows us to isolate terms not recognized by the KSS system. Using these tools we created two reduced feature sets for use in the trials (Table 1). The first, called IQ87, was created by KSS using IQ=87 as the threshold value and resulted in a feature set of 62,106 terms. The second, called IQ57, was created by KSS using IQ=57 as the threshold producing a feature set of 78,165 terms. This feature set was then further reduced to a 33,191 term feature set by removing all abbreviations and terms or names not recognized by the KSS system.

Each document used in the trials was represented by a document vector of individual term TF * IDF values for that document. The vector length for the IQ87 data set was 62,106 and for the IQ57 data set was 33,191.

For each trial we ran both the artificial neural net (ANN) and SVM algorithms on a randomly chosen set of 600 documents from the document collection. Because of the random selection of documents many of the 63 categories had few or no documents assigned (Table 2) and were problematic in terms of interpreting the effects of recall and precision in those categories. Consequently, in addition to analyzing data from the full set of categories we also analyzed the results from only those categories in the test sets to which ten or more documents had been assigned.

Table 1. Test Data Summary

	IQ=57 with unknowns removed	**IQ=87 with unknowns included**
Number of documents	11,327	11,327
Number of categories	63	63
Number of terms	33,191	62,106

Table 2. Testing Data Collections

Trial	**Total Number of Categories**	**Total Number of Documents**	**Number of Categories>=10**	**Documents in Categories >=10**
SVM (IQ87)	63	600	10	503
SVM(IQ57)	63	600	7	478
ANN(IQ87)	63	600	10	501
ANN(IQ57)	63	600	9	486

3.2. Performance Measurements

Classification performance is measured using both recall and precision. In this case, recall is the proportion of the correct documents that are assigned to a category by the algorithm. Precision is the proportion of documents assigned to a category that belong to that category. Text categorization is essentially a series of dichotomous results and so both micro and macro averaging can be used to generate an overall performance over the set of categories used.

In addition we use a single measure, called the F_1 Measure [16], to compare the overall results of the algorithms. The F_1 Measure combines recall and precision with equal weighting and has been used to summarize comparative results.

3.3. Training Set

For the training 600 documents were chosen randomly, without replacement for each trial.

3.4. Algorithms

Both of the algorithms used a *one-on-one* approach in which k(k-1)/2 binary classifiers are created, where k is the number of predefined categories. In the case of text categorization given documents may belong to more than one category and hence the process is a series of binary classifications, i.e., does this document fit into this category or not. This method uses longer training times but provides faster testing times for the SVMs. The complexity analysis for the two algorithms are discussed below and summarized in Table 3.

SVM Algorithm. Each of the 2016 SVM classifiers represents a unique binary combination of the 63 categories. A voting strategy is used in which each SVM votes on which class a given document belongs in. In our case, SVMs were used for the 20 documents from the training set for the 2 categories involved. This processing was done using the LIBSVM [2] module for MatLab [10]. The LIBSVM is an implementation of the SVM classifier algorithm. Furthermore, previous research [12, 16] found that the linear SVM performed as least as well as the non-linear SVMs tested. Joachims [7] found that the SVM produced lower error rates than other classification methods tested. In general, the performance of SVMs is not affected by large dimension problems. The computational complexity for training the SVMs is Nm^2, where N is the number of classifiers and m the number of training examples. The performance is more sensitive to the number of training examples than to the number of classifiers. In practice the training expense is slightly less than Nm^2; Joachims [7] found it to be approximately $Nm^{1.7}$. The complexity for testing is $N\alpha^2$, where $2<\alpha\leq m$, but in practice $\alpha << m$.

Artificial Neural Network Algorithm. The artificial neural network (ANN) algorithm used was based on the perceptron approach, i.e., a neural network without a hidden layer. This approach was chosen as it is easy to construct although it may lead to poorer quality in the classification (Demuth and Beale [5] suggest Self-Organizing Functions are better for classification tasks). As with the SVM model we created a perceptron classifier for each binary combination of categories. The classifiers had an input layer of one node per term. The perceptron had an output layer with one node that produced either a 0 or a 1 indicating which of the two categories a given example belong to. Using the full feature set of terms for the input layer increases the computation required but provides consistency in the feature set for comparison with the SVM algorithm.

In a manner similar to the SVM approach, an ANN was created for each binary combination of categories. For the training phase, each classifier was trained with a set of 20 documents, 10 from each of the two classes that define the classifier. The MatLab Neural Network Toolbox [11] software was used for this processing.

The computational complexity for testing ANNs is V^2N, where V is the number of attributes in the term vector, in our case the number of input nodes, and N is the number of classifiers.

Table 3. Summary of ANN and SVM Complexity

Model	Memory requirement	Computational Complexity
ANN	Matrix multiplication	$O(NV^2)$
SVM	$O(qm)+O(q^2)+O(m)$ Where q is the number of features relevant.	$O(N\alpha^2)$

The ANN algorithm is more sensitive to the size of the term vector than is the SVM algorithm.

4. Results

The trial results are evaluated using standard recall and precision measurements. Recall measures the proportion of the items belonging to a category that are correctly associated with that category. Precision is the proportion of items associated with a category that belong in that category. Both algorithms determine the categorization based on a series of dichotomous classification problems, and so we compute the recall and precision for each and use the average of these individual category recall and precision results (macroaveraging). In addition, for confirmation, we also compute overall average recall and precision over all of the categories using aggregate values (microaveraging) [8].

Many of the categories have few or no documents assigned to them and so we also computed the recall and precision for only those categories with at least 10 documents assigned. Overall the reduced categories, shown in Table 4, included 82% of the 600 test documents.

Table 4. Coverage of Categories > 10

Algorithm	Number of Documents in Categories > 10
SVM87	503
SVM57	478
ANN87	501
ANN57	486
Average	82%

4.1. MacroAveraging Results

For macroaveraging results we computed the recall and precision for each category and averaged the results for each trial, for both the complete and reduced set of categories. The results are given in Table 5.

4.2. MicroAveraging Results

For microaveraging, we aggregated the results over all categories and computed an overall recall and precision for each of the trials, for both the full and reduced set of categories. The micro averaging results are given in Table 6.

4.3. Combined Measure

We calculated the F_1 measure value [16] for both macroaveraged and microaveraged results over all trials, as shown in Table 7. The F_1 measure combines recall and precision values into a single combined measure as follows, were r and p are recall and precision values,

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

Table 5. MacroAveraging Results

		IQ=87		**IQ=57**	
		Recall	**Precision**	**Recall**	**Precision**
SVM					
	All categories	56.98	71.54	61.70	68.07
	Categories≥10	75.27	80.37	81.29	89.50
ANN					
	All categories	57.35	54.79	68.55	67.87
	Categories ≥10	62.18	65.98	58.53	81.58

Table 6. MicroAveraging Results

		IQ=87		**IQ=57**	
		Recall	**Precision**	**Recall**	**Precision**
SVM					
	All categories	78.17	78.17	83.67	83.67
	Categories≥10	83.69	83.04	88.28	94.83
ANN					
	All categories	63.67	63.67	78.50	78.50
	Categories≥10	66.67	71.52	84.16	89.30

Table 7. Combined Measure F_1 Results

	MacroAveraging		MicroAveraging	
Trial	F_1		F_1	
	SVM	**ANN**	**SVM**	**ANN**
IQ87				
all	63.43	56.04	78.17	63.67
cat≥10	77.73	64.02	83.37	69.01
IQ57				
all	64.73	68.21	83.67	78.50
cat≥10	85.19	68.16	91.44	86.65

Table 8. Student t-test one-tail comparison of macroaverage results for categories >10

	$Mean_1$	**$StDev_1$**	**$Mean_2$**	**$StDev_2$**	**p**
SVM IQ87 SVM IQ57 recall	75.27	18.97	81.29	14.87	p<0.01
SVM IQ87 SVM IQ57 precision	80.37	19.33	89.50	19.98	p<0.01
SVM IQ57 ANN IQ57 recall	58.53	28.86	81.29	14.87	p<0.01
SVM IQ57 ANN IQ57 precision	81.58	19.87	89.50	19.98	p<0.01

4.4 Significance of Differences

Finally we were interested in testing for significance in the differences produced by the tests. Student t-tests were run on the macroaverage data in order to compare the significance of the differences between the two SVM trials and between the SVM and ANN trials, using only the reduced feature set. The results of the t-tests are shown in Table 8. In all cases, the one-tailed test results indicate that there is significant difference in the means of these trials, that the SVM algorithm using the reduced feature set, IQ57, is better for both recall and precision than the full feature set, IQ87, and the SVM algorithm performs better than the ANN for this data set for both recall and precision.

5. Conclusion

This data set contains a large number of short documents that populate a largely sparse category set with only a few, about 10% of the categories, having 10 or more documents assigned, accounting for over 80% of the documents.

In the overall comparison of SVM and ANN algorithms for this data set, the results over all conditions for both recall and precision indicate significantly differences in the performance of the SVM algorithm over the ANN algorithm and of the reduced feature set over the larger feature set. Recognizing that SVM is a less (computationally) complex algorithm than the ANN, we conclude that SVM is preferable at least for this genre of data, i.e., many short text documents in a relatively few well populated categories. We say that the SVM algorithm is less complex than ANNs because generally the parameter α that constructs the hyperplane is very small. α can be expected to be especially small because the hyperplane is linear. ANNs on the other hand have to perform large matrix calculations on matrices with as many rows as features. We also found that reducing the vector size by half does not have a negative effect on performance, in fact improves performance, and is therefore also preferable.

Future work includes finer calibration of IQ thresholds and the effect of this threshold on recall and precision levels so that an optimal can be defined for document sets or so that users can set the IQ threshold to tailor the results with respect to precision and recall.

6. Acknowledgements

The authors would like to acknowledge the support of IBM in allowing access to their Textract, KSS and GTR software, used in this research. We would also like to acknowledge the assistance of Dr. James W. Cooper of IBM TJ Watson Research Center in answering our questions about the software. We would also like to thank the Canada Foundation for Innovation for the computer equipment used in this research.

7. References

[1] Burges, C.J.C. (1996). Simplified Support Vector Decision Rules. *13th International Conference on Machine Learning.*

[2] Chang, Chih-Chung, and Chih-Jen Lin (2001), LIBSVM: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/!cjlin/libsvm

[3] Clarke, James. (2002). SGML Parser. Online at:[http://www.jclark.cpm/sp] Available: March, 2002.

[4] Cooper, James W. KSS (Knowledge System Server), A Java class library for Text mining. IBM T J Watson Research Center, Hawthrone, N.Y.

[5] Demuth, H., Beale M. (1998). Neural Network Toolbox User's Guide. The MATH WORKS Inc.

[6] Dumais, S., J.Platt, D.Heckman, and M.Sahami. (1998). Inductive Learning Algorithms and Representations for text Categorization. *7th International Conference on Information and Knowledge Management.*

[7] Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with Many Relevant Features. *Proceedings of ECML-98, 10th European Conference on Machine Learning.*

[8] Kwok, J.T-K. (1998) Automated Text Categorization Using Support Vector Machine. *Proceedings of the International Conference on Neural Information Processing (ICONIP).*

[9] Lewis, D. (2001). Reuters-21578 (ModApte split). Online at: [http://www.daviddlewis.com/resources/testcollections/reuters21578/] Available: September 1997

[10] Ma, Junshui, Zhao, Yi and Stanley Ahalt. *OSU SVM Classifier Matlab Toolbox (ver 3.00)* Online at: [http://eewww.eng.ohio-state.edu/~maj/osu_svm/] Available: September 2002.

[11] Mathworks. Neural Network Toolbox. Online at: [http://www.mathworks.com/access/helpdesk/help/toolbox/nnet/nnet.shtml] Available September 2002.

[12] Rennie, J.D.M. and R. Rifkin. (2001). Improving Multiclass Text Classification with the Support Vector Machine. Online at: [http://www.ai.mit.edu/…] Available: May 23, 2002

[13] Salton, G. and M. McGill. (1983). Introduction to information retrieval. McGraw-Hill, Inc., New York, NY.

[14] Vapnik (1995), *The Nature of Statistical Learning Theory*. Springer, Berlin.

[15] Wemter, S., G. Arevian, and C. Pancjev., (1999). Recurrent Neural Network Learning for Text Routing. *Proceedings of the International Conference on Artificial Neural Networks*. P. 898-903, Edinburgh, UK.

[16] Yang, Y. and X.Liu. (1999). A re-examination of text categorization methods. *Proc. of the ACM SIGIR Conference on Research and Development in Information Retrieval.*