Text Categorization with Support Vector Machines: Learning with Many Relevant Features

Thorsten Joachims

Universität Dortmund
Informatik LS8, Baroper Str. 301
44221 Dortmund, Germany

Abstract. This paper explores the use of Support Vector Machines (SVMs) for learning text classifiers from examples. It analyzes the particular properties of learning with text data and identifies why SVMs are appropriate for this task. Empirical results support the theoretical findings. SVMs achieve substantial improvements over the currently best performing methods and behave robustly over a variety of different learning tasks. Furthermore, they are fully automatic, eliminating the need for manual parameter tuning.

1 Introduction

With the rapid growth of online information, text categorization has become one of the key techniques for handling and organizing text data. Text categorization techniques are used to classify news stories, to find interesting information on the WWW, and to guide a user's search through hypertext. Since building text classifiers by hand is difficult and time-consuming, it is advantageous to learn classifiers from examples.

In this paper I will explore and identify the benefits of *Support Vector Machines (SVMs)* for text categorization. SVMs are a new learning method introduced by V. Vapnik et al. [9] [1]. They are well-founded in terms of computational learning theory and very open to theoretical understanding and analysis.

After reviewing the standard feature vector representation of text, I will identify the particular properties of text in this representation in section 4. I will argue that SVMs are very well suited for learning in this setting. The empirical results in section 5 will support this claim. Compared to state-of-the-art methods, SVMs show substantial performance gains. Moreover, in contrast to conventional text classification methods SVMs will prove to be very robust, eliminating the need for expensive parameter tuning.

2 Text Categorization

The goal of text categorization is the classification of documents into a fixed number of predefined categories. Each document can be in multiple, exactly one, or no category at all. Using machine learning, the objective is to learn classifiers from examples which perform the category assignments automatically. This is a supervised learning problem. Since categories may overlap, each category is treated as a separate binary classification problem.

The first step in text categorization is to transform documents, which typically are strings of characters, into a representation suitable for the learning algorithm and the classification task. Information Retrieval research suggests that word stems work well as representation units and that their ordering in a document is of minor importance for many tasks. This leads to an attribute-value representation of text. Each distinct word[1] w_i corresponds to a feature, with the number of times word w_i occurs in the document as its value. To avoid unnecessarily large feature vectors, words are considered as features only if they occur in the training data at least 3 times and if they are not "stop-words" (like "and", "or", etc.).

This representation scheme leads to very high-dimensional feature spaces containing 10000 dimensions and more. Many have noted the need for feature selection to make the use of conventional learning methods possible, to improve generalization accuracy, and to avoid "overfitting". Following the recommendation of [11], the *information gain* criterion will be used in this paper to select a subset of features.

Finally, from IR it is known that scaling the dimensions of the feature vector with their *inverse document frequency (IDF)* [8] improves performance. Here the "tfc" variant is used. To abstract from different document lengths, each document feature vector is normalized to unit length.

3 Support Vector Machines

Support vector machines are based on the *Structural Risk Minimization* principle [9] from computational learning theory. The idea of structural risk minimization is to find a hypothesis h for which we can guarantee the lowest true error. The true error of h is the probability that h will make an error on an unseen and randomly selected test example. An upper bound can be used to connect the true error of a hypothesis h with the error of h on the training set and the complexity of H (measured by VC-Dimension), the hypothesis space containing h [9]. Support vector machines find the hypothesis h which (approximately) minimizes this bound on the true error by effectively and efficiently controlling the VC-Dimension of H.

SVMs are very **universal learners**. In their basic form, SVMs learn linear threshold function. Nevertheless, by a simple "plug-in" of an appropriate kernel function, they can be used to learn polynomial classifiers, radial basic function (RBF) networks, and three-layer sigmoid neural nets.

One remarkable property of SVMs is that their ability to learn can be **independent of the dimensionality of the feature space**. SVMs measure the complexity of hypotheses based on the margin with which they separate the

[1] The terms "word" and "word stem" will be used synonymously in the following.

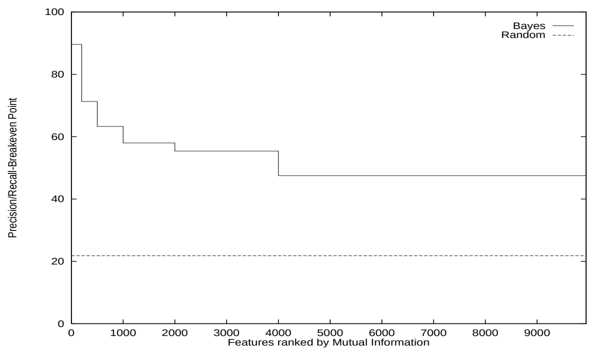

Fig. 1. Learning without using the "best" features.

data, not the number of features. This means that we can generalize even in the presence of very many features, if our data is separable with a wide margin using functions from the hypothesis space.

The same margin argument also suggest a heuristic for **selecting good parameter settings** for the learner (like the kernel width in an RBF network) [9]. The best parameter setting is the one which produces the hypothesis with the lowest VC-Dimension. This allows fully automatic parameter tuning without expensive cross-validation.

4 Why Should SVMs Work Well for Text Categorization?

To find out what methods are promising for learning text classifiers, we should find out more about the properties of text.

High dimensional input space: When learning text classifiers, one has to deal with very many (more than 10000) features. Since SVMs use overfitting protection, which does not necessarily depend on the number of features, they have the potential to handle these large feature spaces.

Few irrelevant features: One way to avoid these high dimensional input spaces is to assume that most of the features are irrelevant. Feature selection tries to determine these irrelevant features. Unfortunately, in text categorization there are only very few irrelevant features. Figure 1 shows the results of an experiment on the Reuters "acq" category (see section 5). All features are ranked according to their (binary) information gain. Then a naive Bayes classifier [2] is trained using only those features ranked 1-200, 201-500, 501-1000, 1001-2000, 2001-4000, 4001-9962. The results in figure 1 show that even features ranked lowest still contain considerable information and are somewhat relevant. A classifier using only those "worst" features has a performance much better than random. Since it seems unlikely that all those features are completely redundant, this leads to the conjecture that a good classifier should combine many features (learn a "dense" concept) and that aggressive feature selection may result in a loss of information.

Document vectors are sparse: For each document, the corresponding document vector contains only few entries which are not zero. Kivinen et al. [4] give both theoretical and empirical evidence for the mistake bound model that "additive" algorithms, which have a similar inductive bias like SVMs, are well suited for problems with dense concepts and sparse instances.

Most text categorization problems are linearly separable: All Ohsumed categories are linearly separable and so are many of the Reuters (see section 5) tasks. The idea of SVMs is to find such linear (or polynomial, RBF, etc.) separators.

These arguments give theoretical evidence that SVMs should perform well for text categorization.

5 Experiments

The following experiments compare the performance of SVMs using polynomial and RBF kernels with four conventional learning methods commonly used for text categorization. Each method represents a different machine learning approach: density estimation using a naive Bayes classifier [2], the Rocchio algorithm [7] as the most popular learning method from information retrieval, a distance weighted k-nearest neighbor classifier [5][10], and the C4.5 decision tree/rule learner [6]. SVM training is carried out with the SVM^{light}[2] package. The SVM^{light} package will be described in a forthcoming paper.

Test Collections: The empirical evaluation is done on two test collection. The first one is the "ModApte" split of the Reuters-21578 dataset compiled by David Lewis. The "ModApte" split leads to a corpus of 9603 training documents and 3299 test documents. Of the 135 potential topic categories only those 90 are used for which there is at least one training and one test example. After preprocessing, the training corpus contains 9962 distinct terms.

The second test collection is taken from the Ohsumed corpus compiled by William Hersh. From the 50216 documents in 1991 which have abstracts, the first 10000 are used for training and the second 10000 are used for testing. The classification task considered here is to assign the documents to one or multiple categories of the 23 MeSH "diseases" categories. A document belongs to a category if it is indexed with at least one indexing term from that category. After preprocessing, the training corpus contains 15561 distinct terms.

Results: Figure 2 shows the results on the Reuters corpus. The *Precision/Recall-Breakeven Point* (see e. g. [3]) is used as a measure of performance and *microaveraging* [10][3] is applied to get a single performance value over all binary classification tasks. To make sure that the results for the conventional methods are not biased by an inappropriate choice of parameters, all four methods were run after selecting the 500 best, 1000 best, 2000 best, 5000 best, (10000 best,) or all features using information gain. At each number of features the values $\beta \in \{0, 0.1, 0.25, 0.5, 1.0\}$ for the Rocchio algorithm and $k \in \{1, 15, 30, 45, 60\}$

[2] http://www-ai.informatik.uni-dortmund.de/thorsten/svm_light.html

					SVM (poly) degree $d =$					SVM (rbf) width $\gamma =$			
	Bayes	Rocchio	C4.5	k-NN	1	2	3	4	5	0.6	0.8	1.0	1.2
earn	95.9	96.1	96.1	97.3	98.2	98.4	**98.5**	98.4	98.3	**98.5**	98.5	98.4	98.3
acq	91.5	92.1	85.3	92.0	92.6	94.6	**95.2**	95.2	95.3	95.0	95.3	95.3	**95.4**
money-fx	62.9	67.6	69.4	78.2	66.9	72.5	75.4	74.9	**76.2**	74.0	75.4	**76.3**	75.9
grain	72.5	79.5	89.1	82.2	91.3	93.1	**92.4**	91.3	89.9	**93.1**	91.9	91.9	90.6
crude	81.0	81.5	75.5	85.7	86.0	87.3	88.6	**88.9**	87.8	**88.9**	89.0	88.9	88.2
trade	50.0	77.4	59.2	77.4	69.2	75.5	76.6	77.3	**77.1**	76.9	78.0	**77.8**	76.8
interest	58.0	72.5	49.1	74.0	69.8	63.3	67.9	73.1	**76.2**	74.4	75.0	**76.2**	76.1
ship	78.7	83.1	80.9	79.2	82.0	85.4	86.0	**86.5**	86.0	**85.4**	86.5	87.6	87.1
wheat	60.6	79.4	85.5	76.6	83.1	84.5	85.2	**85.9**	83.8	**85.2**	85.9	85.9	85.9
corn	47.3	62.2	87.7	77.9	86.0	86.5	85.3	**85.7**	83.9	**85.1**	85.7	85.7	84.5
microavg.	**72.0**	**79.9**	**79.4**	**82.3**	84.2	85.1	85.9	86.2	85.9	86.4	86.5	86.3	86.2
					combined: **86.0**					combined: **86.4**			

Fig. 2. Precision/recall-breakeven point on the ten most frequent Reuters categories and microaveraged performance over all Reuters categories. k-NN, Rocchio, and C4.5 achieve highest performance at 1000 features (with $k = 30$ for k-NN and $\beta = 1.0$ for Rocchio). Naive Bayes performs best using all features.

for the k-NN classifier were tried. The results for the parameters with the best performance on the test set are reported.

On the Reuters data the k-NN classifier performs best among the conventional methods (see figure 2). This replicates the findings of [10]. Compared to the conventional methods all SVMs perform better independent of the choice of parameters. Even for complex hypotheses spaces, like polynomials of degree 5, no overfitting occurs despite using all 9962 features. The numbers printed in bold in figure 2 mark the parameter setting with the lowest VCdim estimate as described in section 3. The results show that this strategy is well-suited to pick a good parameter setting automatically and achieves a microaverage of 86.0 for the polynomial SVM and 86.4 for the RBF SVM. With this parameter selection strategy, the RBF support vector machine is better than k-NN on 63 of the 90 categories (19 ties), which is a significant improvement according to the binomial sign test.

The results for the Ohsumed collection are similar. Again k-NN is the best conventional method with a microaveraged precision/recall-breakeven point of 59.1. C4.5 fails on this task (50.0) and heavy overfitting is observed when using more than 500 features. Naive Bayes achieves a performance of 57.0 and Rocchio reaches 56.6. Again, with 65.9 (polynomial SVM) and 66.0 (RBF SVM) the SVMs perform substantially better than all conventional methods. The RBF SVM outperforms k-NN on all 23 categories, which is again a significant improvement.

Comparing training time, SVMs are roughly comparable to C4.5, but they are more expensive than naive Bayes, Rocchio, and k-NN. Nevertheless, current research is likely to improve efficiency of SVM-type quadratic programming problems. SVMs are faster than k-NN at classification time. More details can found in [3].

6 Conclusions

This paper introduces support vector machines for text categorization. It provides both theoretical and empirical evidence that SVMs are very well suited for text categorization. The theoretical analysis concludes that SVMs acknowledge the particular properties of text: (a) high dimensional feature spaces, (b) few irrelevant features (dense concept vector), and (c) sparse instance vectors.

The experimental results show that SVMs consistently achieve good performance on text categorization tasks, outperforming existing methods substantially and significantly. With their ability to generalize well in high dimensional feature spaces, SVMs eliminate the need for feature selection, making the application of text categorization considerably easier. Another advantage of SVMs over the conventional methods is their robustness. SVMs show good performance in all experiments, avoiding catastrophic failure, as observed with the conventional methods on some tasks. Furthermore, SVMs do not require any parameter tuning, since they can find good parameter settings automatically. All this makes SVMs a very promising and easy-to-use method for learning text classifiers from examples.

References

1. C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, November 1995.
2. T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *International Conference on Machine Learning (ICML)*, 1997.
3. T. Joachims. Text categorization with support vector machines: Learning with many relevant features. Technical Report 23, Universität Dortmund, LS VIII, 1997.
4. J. Kivinen, M. Warmuth, and P. Auer. The perceptron algorithm vs. winnow: Linear vs. logarithmic mistake bounds when few input variables are relevant. In *Conference on Computational Learning Theory*, 1995.
5. T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
6. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
7. J. Rocchio. Relevance feedback in information retrieval. In G. Salton, editor, *The SMART Retrieval System: Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall Inc., 1971.
8. G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.
9. Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 1995.
10. Y. Yang. An evaluation of statistical approaches to text categorization. Technical Report CMU-CS-97-127, Carnegie Mellon University, April 1997.
11. Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *International Conference on Machine Learning (ICML)*, 1997.