
Evaluation of the Highest Probability SVM Nearest Neighbor Classifier with Variable Relative Error Cost

Enrico Blanzieri
University of Trento, Italy
blanzier@dit.unitn.it

Anton Bryl
University of Trento, Italy;
Create-Net, Italy
abryl@dit.unitn.it

Abstract

In this paper we evaluate the performance of the highest probability SVM nearest neighbor (HP-SVM-NN) classifier, which combines the ideas of the SVM and k -NN classifiers, on the task of spam filtering. To classify a sample, the HP-SVM-NN classifier does the following: for each k in a predefined set $\{k_1, \dots, k_N\}$ it trains an SVM model on k nearest labeled samples, uses this model to classify the given sample, and transforms the output of SVM into posterior probabilities of the classes using sigmoid approximation; then it selects that of the $2 \times N$ resulting answers which has the highest probability. The evaluation shows that in terms of ROC curves the algorithm is able outperform pure SVM.

1 Introduction

The problem of unsolicited bulk email, or *spam*, is today well-known to every user of the Internet. Spam not only causes misuse of time and computational resources, thus leading to financial losses, but it is also often used to advertise illegal goods and services or to promote online frauds. A widely applied way of anti-spam protection is spam filtering. A great number of learning-based spam filters are proposed in the literature. Some of them use the knowledge about the structure of the message header, retrieving particular kinds of technical information and classifying messages according to it, for example the method based on SMTP path analysis [11]. Other methods use human language technologies to analyze the message content, for example the approach based on smooth n -gram language modelling [12]. However, there is a large group of learning-based filters that observe a message just as a set of tokens. The most popular method in this group is Naïve Bayes [15]. The Support Vector Ma-

chine (SVM) classifier was proposed for spam filtering by Drucker et al. [7] and showed good results in comparison to other methods [7, 10, 16], which makes it a reasonable quality baseline. We must also mention the approaches based on maximum entropy model [16] and boosting [7], both comparable in accuracy to SVM [16]. A filter based on the k -nearest neighbor (k -NN) algorithm was introduced by Androutsopoulos et al. [1], and showed comparatively low results [10, 16].

In this paper we evaluate the performance of the highest probability SVM nearest neighbor classifier, which is an improvement over the SVM nearest neighbor classifier, on the task of spam filtering. SVM nearest neighbor (SVM-NN) [4] is a combination of SVM and k -NN. In order to classify a sample x , it first selects k training samples nearest to the sample x , and then uses this k samples to train an SVM model which is further used to make the decision. This method is able to achieve a smaller generalization error bound in comparison to pure SVM because of a bigger margin and a smaller ball containing the points. The motivation for using this classifier for spam filtering is the following. Spam is not uniform, but rather consists of messages on different topics [8] and in different genres [6]. The same can be applied also to legitimate mail. This suggests that a classifier which works on a local level can achieve good results on this data. As such, the SVM-NN algorithm proposes no rule for selecting the parameter k . Blanzieri and Bryl [3] made an attempt to estimate k by internal training and testing on the training data, but this approach brought uncertain results. Instead, with the highest probability SVM-NN (HP-SVM-NN), we propose to select the parameter k which minimizes the posterior probability of error. The probability of error is estimated from the output of SVM using the sigmoid approximation [14]. The description and some preliminary experimental evaluation of the method is given in our technical report [2]. In particular, we showed that with equal error cost the proposed method is able to outperform pure SVM. Here we present further experimental evaluation

of HP-SVM-NN, paying attention to the possibility to adjust the balance between the two types of errors. The experiments show that the proposed algorithm is able to outperform pure SVM. We also discuss a way of building a practical spam filter based on this classifier. In fact, the low speed of the classifier together with its high accuracy suggest that it is reasonable to use it not as a separate filter, but in cascade with a faster algorithm, for example Naïve Bayes.

The rest of the paper is organized as follows. In Section 2 we describe the algorithm, Section 3 is dedicated to the experimental evaluation, Section 4 addresses building a practical filter, and Section 5 is a conclusion.

2 The Algorithm

To present the highest probability SVM nearest neighbor classifier, we need first to describe the SVM and the SVM nearest neighbor classifiers.

2.1 Support Vector Machines

Support Vector Machine (SVM) is a state of the art classifier [5]. Below we describe it briefly. Let there be n labeled training samples from two classes. Each sample x_i is a vector of dimensionality d , each label y_i is either 1 or -1 depending on the class of the sample. Thus, the training data can be described as follows:

$$T = ((x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)), x_i \in \mathbb{R}^d, y_i \in \{\pm 1\}.$$

Given this training samples and a predefined transformation $\Phi : \mathbb{R}^d \rightarrow F$, which maps the features to a transformed feature space, the classifier builds a decision rule of the following form:

$$y_p(x) = \text{sign} \left(\sum_{i=1}^L \alpha_i y_i K(x_i, x) + b \right),$$

where $K(a, b) = \Phi(a) \cdot \Phi(b)$ is the kernel function, and α_i and b maximize the margin of the separating hyperplane. The result can be got also not in the binary form, but in the form of a real number (which is proportional to the signed distance from the sample to the separating hyperplane), by dropping the sign function:

$$y'_p(x) = \sum_{i=1}^L \alpha_i y_i K(x_i, x) + b,$$

Using an additional parameter it is also possible to consider the unequal error cost with SVM [13].

Some applications require not a binary classification decision, but posterior probabilities of the classes $P(\text{class}|\text{input})$. SVM provides no direct way to obtain such probabilities, but there exist several ways to

approximate them. In particular, Platt [14] proposed to use sigmoid approximation:

$$P(y = 1|y'_p) = \frac{1}{1 + e^{Ay'_p + B}} \quad (1)$$

where A and B are the parameters obtained by fitting on an additional training set. Platt observes, that for the SVM classifier with the linear kernel it is acceptable to use the same training set both for training the SVM model and for fitting the sigmoid. The description of the procedure of fitting the parameters A and B can be found in the Platt's original publication [14].

2.2 The SVM Nearest Neighbor Classifier

The SVM Nearest Neighbor (SVM-NN) classifier [4] combines the ideas of SVM and k -NN. To classify a sample x , the algorithm selects k samples nearest to the sample x , and uses this k samples to train an SVM model and perform the classification. Samples nearest to the sample x in the transformed feature space are those with minimal values of $K(x_i, x_i) - 2K(x_i, x)$, as can be seen from the following equality:

$$\begin{aligned} \|\Phi(x_i) - \Phi(x)\|^2 &= \Phi^2(x_i) + \Phi^2(x) - 2\Phi(x_i) \cdot \Phi(x) = \\ &= K(x_i, x_i) + K(x, x) - 2K(x_i, x) \end{aligned}$$

A problem is that this algorithm, as such, provides no procedure of finding an appropriate value for the parameter k . In previous works the parameter k was determined by means of a validation set [3, 4]. The approach proved to be useful for remote sensing data, where the procedure outperformed SVM, but not for spam filtering, where the results were not conclusive. In both cases the SVM-NN methods outperformed the plain k -NN classifier based on the majority vote.

2.3 The Highest Probability SVM Nearest Neighbor Classifier

The highest probability SVM Nearest Neighbor (HP-SVM-NN) classifier is based on the idea of selecting the parameter k from a predefined set $\{k_1, \dots, k_N\}$ separately for each sample x which must be classified. To do this, the classifier first performs the following actions for each considered k : k training samples nearest to the sample x are selected; an SVM model is trained on this k samples; then, the same k samples are classified using this model, and the output is used to fit the parameters A and B in the equation (1); then, the sample x is classified using this model, and the estimates of the probabilities $P(\text{legitimate}|x)$ and $P(\text{spam}|x) = 1 - P(\text{legitimate}|x)$ are calculated; in this way, $2 \times N$ answers are obtained. Then, the answer with the highest posterior probability estimate is chosen. An additional parameter C can be used to

Algorithm: The Highest Probability SVM Nearest Neighbor Classifier

Require: sample x to classify; training set $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$;
set of possible values for the number of nearest neighbors $\{k_1, k_2, \dots, k_N\}$;
parameter C for adjusting the balance between the two types of errors.

Ensure: decision $y_p \in \{-1, 1\}$

- 1: Order the training samples by the value of $K(x_i, x_i) - 2 * K(x_i, x)$ in ascending order.
- 2: MinErr = 1000; Value = 0
- 3: **if** the first k_1 values are all from the same class c **then**
- 4: **return** c
- 5: **end if**
- 6: **for all** k **do**
- 7: Train SVM model on the first k training samples in the ordered list.
- 8: Classify x using the SVM model with equal error costs, get the result y'_p
- 9: Classify the same training samples using this model.
- 10: Fit the parameters A and B for the estimation of $P(y = 1|y'_p)$.
- 11: ErrorNegative = $P(y = 1|y'_p)$; ErrorPositive = $1 - P(y = 1|y'_p)$
- 12: **if** ErrorPositive < MinErr **then**
- 13: MinErr = ErrorPositive; Value = 1
- 14: **end if**
- 15: **if** ErrorNegative * C < MinErr **then**
- 16: MinErr = ErrorNegative * C ; Value = -1
- 17: **end if**
- 18: **end for**
- 19: **return** Value

Figure 1: The Highest Probability SVM Nearest Neighbor Classifier: pseudocode.

adjust the balance between the two types of errors. In this case, the probability of error for the negative answer must be not just lower than the probability of error for the positive answer, but at least C times lower to be selected. If false positives are less desirable than false negatives, $C < 1$ should be used

We must mention, that from the point of view of speed such algorithm is not the same as N runs of basic SVM-NN classifier, because the costly operation of distance calculation is performed only once for each classified sample. Nevertheless, in this form the algorithm is very slow and needs some optimization to allow practical usage or fast experimental evaluation. Such optimization is possible because for some samples with the smallest considered k all the nearest neighbors selected are from the same class. In this degenerated case the estimate of posterior probability of the class from which the neighboring samples come is equal to 1.0, and so cannot be exceeded. If such case occurs, the decision is taken immediately and no further search is performed. This version of the algorithm is faster than the initial one. The pseudocode of this optimized version of the algorithm is presented on Figure 1.

3 Experimental Evaluation

In order to evaluate the performance of the proposed algorithm and to compare it with pure SVM, we established an experiment using ten-fold cross-validation on the SpamAssassin corpus¹. The corpus consists

of 4150 legitimate messages and 1897 spam messages. The partitioning of the data is the same for all the runs. The linear kernel was used in both classifiers. The following set of 12 possible values of the parameter k was used: {50, 150, 250, 350, 500, 700, 1000, 1400, 2000, 2800, 4000, 5400}.

Feature extraction is performed as follows. Each part of the message, namely the message body and each field of the message header, is considered as an unordered set of strings (tokens) separated by spaces. Presence of a certain token in a certain part of the message is considered a binary feature of this message. Then, the d features with the highest information gain are selected. Information gain is defined as follows:

$$IG(f_k) = \sum_{c \in \{c_1, c_2\}} \sum_{f \in \{f_k, \bar{f}_k\}} P(f, c) \times \log \frac{P(f, c)}{P(f) \times P(c)},$$

where f_k is a binary feature, and c_1 and c_2 are the two classes of samples. Thus, each message is represented with a vector of d binary features.

In order to build ROC curves, for the HP-SVM-NN classifier the parameter C (see Figure 1) is changed, and for the SVM classifier the balance between the two types of errors is changed by modifying the relative error cost parameter. For all the other parameters default values are used. Inside HP-SVM-NN the SVM classifier is always used with default parameters. The implementation of SVM used by us is *SVMlight*² [9]. Feature extraction is performed by a Perl script. The

¹<http://spamassassin.apache.org/publiccorpus/>

²<http://svmlight.joachims.org/>

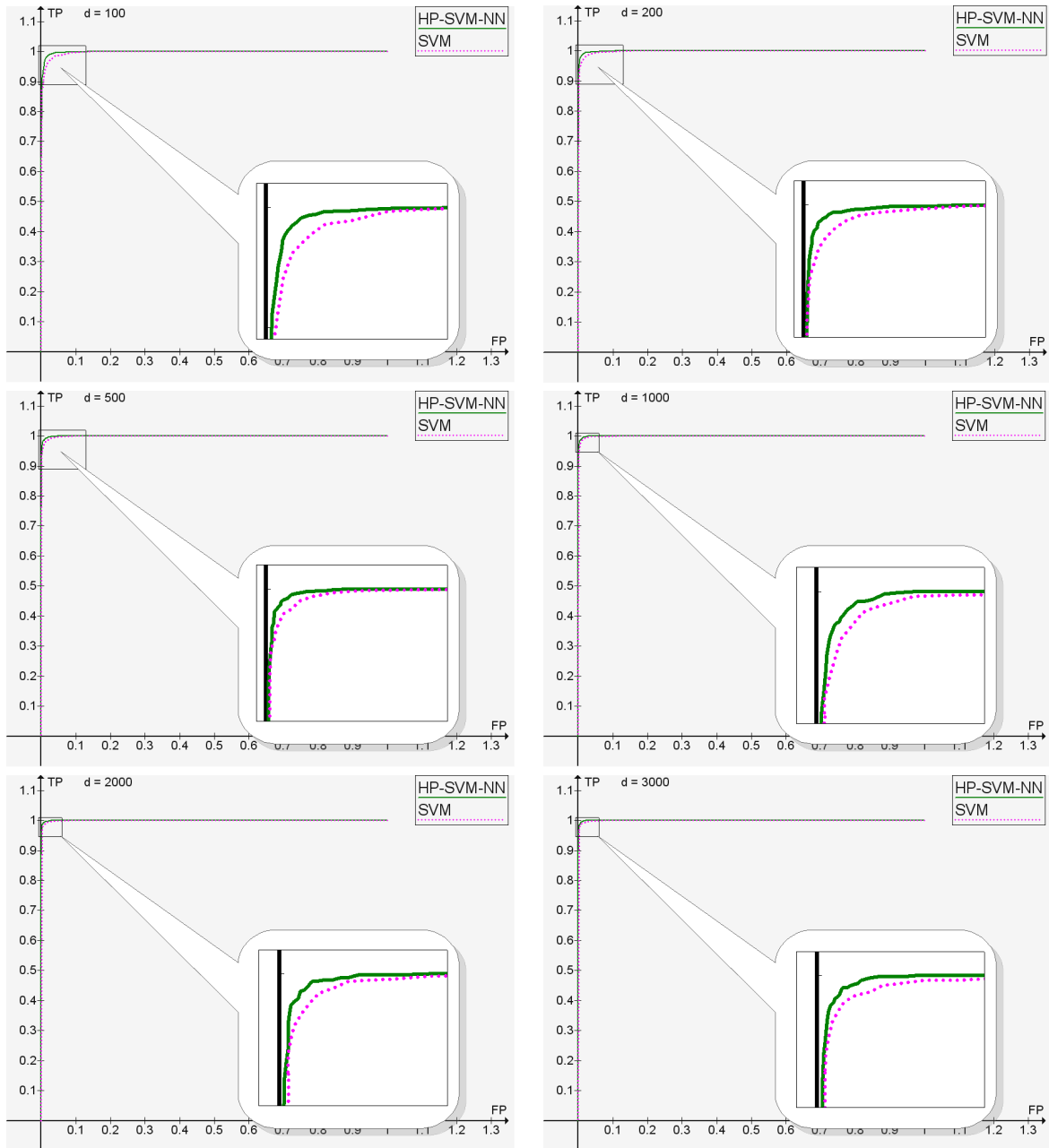


Figure 2: Comparison of SVM and HP-SVM-NN in terms of ROC curves. *SVM* is the pure SVM classifier, as implemented in *SVMlight*. *HP-SVM-NN* is the highest probability SVM-NN classifier, as described in section 2.3. d is the number of features. Positive class is spam, negative class is legitimate mail.

HP-SVM-NN classifier is implemented as a Perl script which uses *SVMlight* utilities as external modules for SVM training and classification.

Figure 2 shows the comparison of ROC curves for SVM and HP-SVM-NN³. We can see that HP-SVM-NN performs better with all the numbers of features considered. Since both methods have high accuracy, the dif-

ference between the curves may seem small. However, having true positive rate, for example, of 99% instead of 98% means in fact twice less spam in one's mailbox.

A disadvantage of our classifier is its low speed. Our implementation, with the number of features $d = 500$, number of possible values of k equal to 12 and the training dataset of about 5400 samples, classifies an “easy” sample (with degenerate neighborhood) in about a second and a usual sample in about ten sec-

³In the initial version of the paper the figure for $d = 200$ was plotted wrongly. This is a corrected version (14.11.07).

onds on a PC with CPU speed of 2.50GHz. However, there is much space for optimization at the software level, by which improvement on speed can be achieved.

4 Building a Hybrid Filter

The low speed of the HP-SVM-NN classifier suggests that it is reasonable not to build a filter based on this classifier alone, but to combine it with a faster algorithm, for example Naïve Bayes, so that only “hard” messages are classified by HP-SVM-NN. For example, the following procedure can be used to classify a message. First, a fast classifier with very high cost of false positives is applied to the message, and if it is classified as spam, no further check is performed. Otherwise, this decision is re-checked by the same classifier, but this time with very high cost of false negatives. If the answer is again “non-spam”, this decision is final, else the HP-SVM-NN classifier is used to make the final decision. In this way, only a small subset of the data will be classified with the HP-SVM-NN algorithm, and thus the filter will profit from its high accuracy without loosing much in classification speed.

5 Conclusion

In this paper we evaluated the highest probability SVM nearest neighbor (HP-SVM-NN) classifier applied to the task of spam filtering with variable relative error cost. HP-SVM-NN is a local SVM classifier, which uses k samples in the neighborhood of the sample which must be classified, with the parameter k selected among a pool of values dynamically, depending on the posterior probabilities of the classes estimated as proposed by Platt [14]. Experimental evaluation shows that our classifier is able to outperform pure SVM. Thus locality proved to be a viable way of increasing the accuracy of the classification at the price of extra computation. A way of building a practical filter based on this classifier is discussed. The proposed filter architecture needs evaluation on other datasets. Such evaluation is subject to the nearest future work.

References

- [1] I. Androutsopoulos, G. Paliouras, V. Karkaletsis, G. Sakkis, C. Spyropoulos, and P. Stamatopoulos. Learning to filter spam e-mail: A comparison of a naive bayesian and a memory-based approach. In *Proceedings of the MLTIA Workshop, PKDD 2000*, pages 1–13, 2000.
- [2] E. Blanzieri and A. Bryl. Highest probability SVM nearest neighbor classifier for spam filtering. Technical report #DIT-07-007. 2007.
- [3] E. Blanzieri and A. Bryl. Instance-based spam filtering using SVM nearest neighbor classifier. In *Proceedings of FLAIRS-20*, pages 441–442, 2007.
- [4] E. Blanzieri and F. Melgani. An adaptive SVM nearest neighbor classifier for remotely sensed imagery. In *Proceedings of 2006 IEEE IGARSS*, 2006.
- [5] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
- [6] W. Cukier, S. Cody, and E. Nesselroth. Genres of spam: Expectations and deceptions. In *Proceedings of HICSS '06*, volume 3, 2006.
- [7] H. Drucker, D. Wu, and V. Vapnik. Support vector machines for spam categorization. *IEEE Transactions on Neural networks*, 10(5):1048–1054, 1999.
- [8] G. Hulten, A. Penta, G. Seshadrinathan, and M. Mishra. Trends in spam products and methods. In *Proceedings of CEAS'2004*, 2004.
- [9] T. Joachims. *Making large-Scale SVM Learning Practical*. MIT-Press, 1999.
- [10] C.-C. Lai and M.-C. Tsai. An empirical performance comparison of machine learning methods for spam e-mail categorization. *Hybrid Intelligent Systems*, pages 44–48, 2004.
- [11] B. Leiba, J. Ossher, V. T. Rajan, R. Segal, and M. Wegman. SMTP path analysis. In *Proceedings of CEAS'2005*, 2005.
- [12] B. Medlock. An adaptive approach to spam filtering on a new corpus. In *Proceedings of CEAS'2006*, 2006.
- [13] K. Morik, P. Brockhausen, and T. Joachims. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proceedings 16th International Conference on Machine Learning*, pages 268–277. Morgan Kaufmann, San Francisco, CA, 1999.
- [14] J. Platt. Probabilities for SV machines. In A. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74, 2000.
- [15] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 Workshop*, 1998.
- [16] L. Zhang, J. Zhu, and T. Yao. An evaluation of statistical spam filtering techniques. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(4):243–269, 2004.