Speeding Up the Decision Making of Support Vector Classifiers

Jonathan Milgram – Mohamed Cheriet – Robert Sabourin
École de Technologie Supérieure, Montréal, Canada
milgram@livia.etsmtl.ca – {mohamed.cheriet – robert.sabourin}@etsmtl.ca

Abstract

In this paper, we propose a new approach for speeding up the decision making of Support Vector Classifiers (SVC) in the context of multi-class classification. A two-stage system embedded within a probabilistic framework is presented. In the first stage we pre-estimate the posterior probabilities with a model-based approach and we re-estimate only the highest probabilities with appropriate SVCs in the second stage. We have tested our system on the benchmark database MNIST and the results show that our dynamic classification process allows to speedup the full "pairwise coupling" SVCs by a factor of 7.7 while preserving the accuracy. In addition, although the "one against all" strategy estimate slightly betters probabilities, our modular architecture seems more adapted to large multi-class problems.

1. Introduction

A recent benchmarking of state-of-the-art techniques for handwritten digit recognition [8] has shown that Support Vector Classifier (SVC) gives higher accuracy than classical neural classifiers like Multi Layer Perceptron (MLP) or Radial Basis Function (RBF) networks. However, thanks to the improvement of the computing power and the development of new learning algorithms, it is now possible to train SVC in real world applications. Unfortunately, the test phase of SVC is extremely expensive in terms of computation time, because it is necessary to evaluate the distance of a large number of training examples called support vectors (SVs). This is an important burden for real-time applications. Thus, in this paper we propose to speed up the decision making of SVCs in the context of multi-class problems.

Several approaches to overcome this burden have been suggested in the literature. In order to improve the speed in test phase of each SVC, a solution is to approximate the decision surface of support vectors. Thus, in [2] the goal is then to choose the smallest reduced set vectors, such that any resulting loss in generalization performance remains acceptable. The authors used an unconstrained conjugate gradient method to find the reduced set. Although this method allows to achieve a good speedup in test phase by reducing the complexity of each SVC, the classifying cost remain a problem. By different means, an incremental procedure for growing SVC is proposed in [10]. This approach allows fast classification by sequential evaluation of SVs. The key point is that some regions of the input space can be correctly classified using only a highly reduced subset of the SVs. In addition, an MLP-SVC combination architecture is suggested in [1]. The authors used a pairwise strategy to decompose the multi-class problem in binary sub-problems. An ensemble of specialized SVCs is constructed and during the classification, an MLP is used to select the "good" SVC that will be used to make decision. Thus, this architecture is based on the idea that the correct class almost systematically belongs to the two maximum MLP outputs.

In this paper, we take up the idea of two-stage classification system to reduce the classifying cost of an ensemble of SVCs. Unlike the architecture proposed in [1], we don't always use one SVC and only one. Indeed, with the same idea that some regions of the input space can be easily classified, we extend the notion of dynamic test process proposed in [10] to multi-class problems and we propose to fix the number of SVC used according to the outputs of the first stage. Furthermore, we privileged a modular approach which makes an accurate classification possible when the number of classes is large. Indeed, as it is shown in [9] a global approach like MLP network is not effective to resolve classification problems like Korean characters on postal address (352 classes). Thus, we adopted in the first stage of our system a model-based approach similar to the MQDF suggested in [6] which is low cost and achieved good results for Chinese character recognition where the number of classes was 927. Moreover, we embed our system within a probabilistic framework, because as mentioned in [11]: "The output of a classifier should be a calibrated posterior probability to enable post-processing". Indeed, this type of confidence measure is essential in many application, when the classifier only contributes a small part of the final decision or if it is preferable to make no decision when the result of classification is uncertain. So, in the first stage, we pre-estimate the probabilities with a model-based approach and re-estimate only the highest probabilities with appropriate SVCs in the second stage.

To evaluate our method, we chose a classical pattern recognition problem: isolated handwritten digit recognition. Thus, in our experiments, we used a well-known benchmark database that is available at http://yann.lecun.com/exdb/mnist/. The Modified NIST dataset (MNIST) was extracted from the NIST special

database SD3 and SD7. The original binary images were normalized into 20×20 gray-scale images with preserved aspect ratio and the normalized images were centered by center of mass in 28×28 images. On the other hand, to compare the quality of the probabilities estimate by the different methods, we use the Chow's rule to evaluate their error-reject tradeoff. Indeed, as it is shown in [4] this rule provides the optimal error-reject tradeoff only if the posterior probabilities of the data classes are exactly known. But, in real applications, such probabilities are affected by significant estimate errors. Thus, the better the probabilities estimate is, the better the error-reject tradeoff is.

2. Support Vector Classifier

The training and testing of all SVCs are performed with the LIBSVM software of which all the algorithms are described in [3]. We use the C-SVC with a Gaussian kernel $K(x_k,x)=\exp(-\gamma\|x_k-x\|^2)$. The penalty parameter C and the kernel parameter γ are empirically optimized. The learning dataset has been divided into two subsets. The first 50,000 samples have been used for training and the next 10,000 for validation. Then, we have chosen parameters that minimize the error rate on the validation dataset (1.47 %) with a pairwise coupling and a voting rule. Finally, for all subsequent experiments we used C = 10 and γ = 0.0185.

2.1. Estimate posterior probability

If an SVC can possibly make good decisions, these output values are uncalibrated. But, a simple solution is proposed in [11] to map the SVC outputs into posterior probabilities. Given a training set of instance-label pairs $\{(x_k,y_k):k=1,\ldots,n\}$, where $x_k\in R^d$ and $y_k\in\{1,-1\}$, the unthresholded output of an SVC is

$$f(x)=\sum_{k=1}^{n}y_k\alpha_k K(x_k,x)+\beta \quad , \qquad (1)$$

where the samples with non-zero Lagrange multiplier α_k are called support vectors (SVs).

Since the class-conditional between the margins are apparently exponential the authors suggest to fit an additional sigmoid function (eq. 2) to estimate probabilities.

$$\hat{P}(y=1\mid x)=\frac{1}{1+\exp(af(x)+b)} \qquad (2)$$

The parameter a and b are derived by minimizing the negative log likelihood of the training data, which is a cross-entropy function:

$$-\sum_{k=1}^{n}\left(t_k\log\left(\hat{P}(y_k=1\mid x_k)\right)+(1-t_k)\log\left(1-\hat{P}(y_k=1\mid x_k)\right)\right), \qquad (3)$$

where $t_k=\frac{y_k+1}{2}$ denotes the probability target.

Then, to solve this optimization problem, the author uses a model-trust minimization algorithm based on the Levenberg-Marquardt algorithm. But, in a recent note [7] it is shown that there are two problems in the pseudo-code provided in [11]. One is the calculation of the objective value, and the other is the implementation of the optimization algorithm. Thus, the authors propose another minimization algorithm based on a simple Newton's method with backtracking line search. As we can see in the next sections, we tested the two algorithms on our data and noticed that the later approach induces a bias in the estimation of posterior probabilities.

2.2. One against all

SVC is a binary classifier, so it is necessary to combine several SVCs to solve a multi-class problem. A classical method is the "one against all" strategy in which one SVC per class is constructed. Each classifier is trained to distinguish the examples in a single class from the examples in all remaining classes. Therefore, to estimate posterior probabilities, it is possible to separately optimize each sigmoidal function. Then, the posterior probability of the j th class is obtain by

$$\hat{P}(\omega_j\mid x)=\frac{1}{1+\exp(a_j f_j(x)+b_j)} \quad , \qquad (4)$$

where $f_j(x)$ denotes the output of SVC "j against all". But, nothing guarantees that the sum of all probabilities $\sum_{j=1}^{c}\hat{P}(\omega_j\mid x)$ is equal to one. In fact, we have obtained on the MNIST validation dataset a mean of 0.98 and a standard deviation of 0.11 with the sigmoidal functions optimized by the method proposed in [7]. Thus, with the objective to exploit the outputs of all SVCs to estimate probabilities overall, we propose to use the *softmax* function (eq. 5) which can be regarded as a generalization of the sigmoidal function for multi-class case.

$$\hat{P}(\omega_j\mid x)=\frac{\exp(\alpha f_j(x))}{\sum_{j'=1}^{c}\exp(\alpha f_{j'}(x))} \qquad (5)$$

Then, we re-define for multi-class problem the cross-entropy function, which takes form

$$-\sum_{j=1}^{c}\sum_{k=1}^{n}\left(t_k\log\left(\hat{P}(\omega_j\mid x_k)\right)+(1-t_k)\log\left(1-\hat{P}(\omega_j\mid x_k)\right)\right), \qquad (6)$$

and we optimize the α parameter on the training samples. We tested on validation dataset and obtained the smallest cross-entropy error with *softmax* function (1,126.5 *vs.* 1,250.0 with sigmoid). Moreover, the error-reject tradeoff shown in Fig.1 confirms that a global *softmax* function estimates probabilities slightly better than several local

sigmoidal functions. In addition, Fig.1 shows that the algorithm proposed in [11] is not accurate.

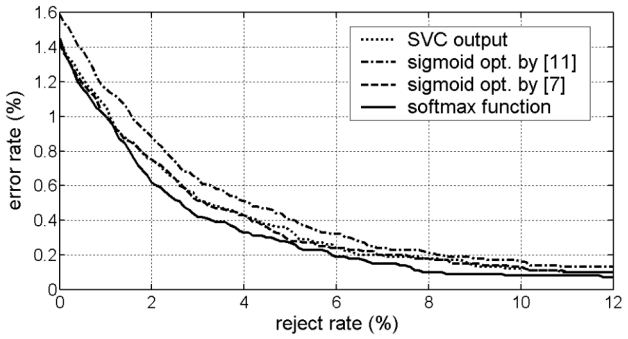

Figure 1. Error-reject tradeoff of "one against all" on the validation dataset

Finally, we obtain on test dataset an error rate of 1.41 % that is comparable with those reported in [2] and [8] when no artificial training examples are generated and no discriminative features are extracted. Hence, we will refer to this first approach for the next experiments. Furthermore, we adopt the number of kernel evaluation per pattern (KEPP) as a measure for the classifying cost, since it is the main cause of the computation effort during the test phase. Thus, the "one against all" strategy requires 13 743 KEPP to make decision.

2.2. Pairwise coupling

Another method for multi-class classification is the "pairwise coupling" strategy, which consists to construct a classifier for each pair of classes. Thus, a *c* class problem is decomposing in *c*(*c*-1)/2 binary sub-problems. Generally, a voting rule can be use to make decision, but we want to estimate posterior probabilities. With this intention, we apply the "Resemblance Model" proposed in [5] to combine posterior probability of each pairwise classifier into posterior probability of multi-class classifier. Then, since prior probabilities are all the same, posterior probabilities can be estimated by

$$\hat{P}(\omega_j \mid x) = \frac{\prod_{j' \neq j} \hat{P}(\omega_j \mid x \in \omega_{j,j'})}{\sum_{j''=1}^{c} \prod_{j' \neq j''} \hat{P}(\omega_{j''} \mid x \in \omega_{j'',j'})}, \quad (7)$$

where $\omega_{j,j'}$ denotes the union of classes ω_j and $\omega_{j'}$.

Furthermore, as reported in [3], although we have to train as many as c(c-1)/2 classifiers, as each problem is easier, the total training time of "pairwise coupling" may not be more than that of the "one against all" method. Indeed, in our experiments, the learning of the 45 "pairwise coupling" SVCs is ten times faster than the learning of only 10 "one against all" SVCs.

In terms of classifying cost, the "pairwise coupling" strategy is slightly lighter than that of "one against all" (11,118 KEPP *vs.* 13,743 KEPP). Moreover, as we can see in Fig.2, firstly, the algorithm proposed in [11] is still not satisfying, and secondly, the "one against all" strategy with a *softmax* function estimates probabilities slightly better than that of "pairwise coupling".

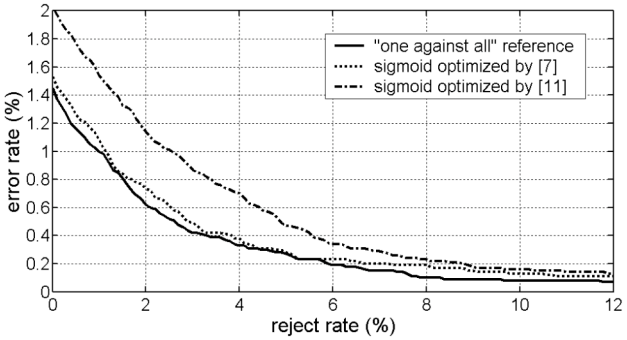

Figure 2. Error-reject tradeoff of "pairwise coupling" on the validation dataset

Finally, we obtained on the test dataset an error rate of 1.48 % that is slightly better than that obtained with a voting rule (1.54 %), but slightly worse than that obtained with the "one against all" strategy (1.41 %).

3. Two-stage classification system

Although the "one against all" strategy is slightly more accurate, since our objective is to speedup the decision making, it seems better to use in the second stage of our system a "pairwise coupling" approach, which is more modular. Indeed, the idea of our two-stage classification system is to pre-estimate the posterior probabilities with a light classification approach and to reduce the number of the plausible classes, for which we re-estimate the posterior probabilities with appropriate SVCs. Thus, if we use "one against all" SVCs in the second stage, we are obliged to calculate the distances of a large number of SVs belonging to the implausible classes, which increases the classifying cost.

3.1. Pre-estimation of posterior probabilities with a model-based approach

At the first stage of our system, we use a simple method to model each class ω_j with a hyperplane defined by the mean vector μ_j, and the matrix Ψ_j of the k eigenvectors of the covariance matrix Σ_j with the largest eigenvalues. Thus, given a data point x of the feature space, the class membership can be evaluated by the distance d_j from the point x to its projection $f_j(x)$ on the hyperplane.

$$d_j(x) = \left\| x - f_j(x) \right\| \quad (8)$$

$$f_j(x) = (x - \mu_j)\Psi_j\Psi_j^{\mathrm{T}} + \mu_j \quad (9)$$

This method required the optimization of only one parameter: the number k of eigenvectors used. However, this parameter strongly influences the accuracy of the classification, as we shown Fig.3.

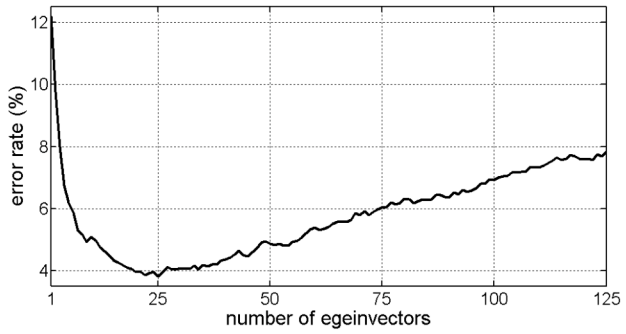

Figure 3. Effect of the dimensionality of the hyperplane models

If k is too small, the models are not precise so we loose too much information. Indeed, while $k = 0$, each class is model by a simple prototype that is the mean vector μ_j of training data. On the other hand, if the value of k is too large, the models are not discriminative. At worst, if $k = d$, where d is the dimension of the input pattern, the hyperplane embeds all the points of the feature space. Hence, for all point x, the projection distance will be null.

Thereafter, we can observe that the distribution of the projection distances between the margins is apparently exponential. Thus, as for the "one against all" SVCs, we use the *softmax* function to map projection distance to posterior probability:

$$\hat{P}_f(\omega_j \mid x) = \frac{\exp(-\alpha\, d_j(x))}{\sum_{j'=1}^{c} \exp(-\alpha\, d_{j'}(x))} \qquad (10)$$

We optimize the α parameter on the training samples and notice in Fig.4 that the use of the *softmax* function improves significantly the error-reject tradeoff of the model-based approach.

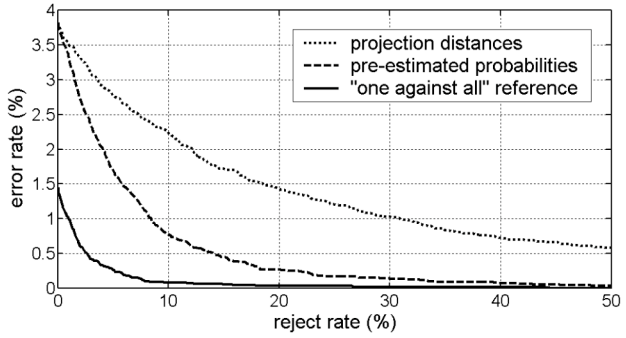

Figure 4. Error-reject tradeoff of the model-based approach on the validation dataset

Moreover, we can see that half of the examples with the highest confidence levels are correctly classified. Finally, we obtain on test dataset an error rate of 4.09 %, which is not satisfying, but still comparable to the 3.34 % obtained by the nearest-neighbor rule.

3.2. Re-estimation of posterior probabilities with Support Vector Classifiers

Thereafter, we determine the list of p classes $\{\omega_{\ell(1)}, \ldots, \omega_{\ell(p)}\}$ of which the posterior probabilities estimated in the first stage are higher than a threshold ε. Hence, $\ell(j)$ is the index of the j th class that verifies $\hat{P}_f(\omega_{\ell(j)} \mid x) > \varepsilon$. Then, if p is superior to one, we use in the second stage the appropriate SVCs to re-estimate the posterior probabilities of the p classes.

In consequence, the final probabilities are not homogeneous, since they can be estimated by different approaches. However, it is not an important drawback. Indeed, when p is superior to one, the first stage estimates only the smallest probabilities, which are negligible, and in this case the second stage estimates all the remaining probabilities. These p significant probabilities are obtained by

$$\hat{P}_s(\omega_{\ell(j)} \mid x) = \frac{\prod_{j''=1, j'' \neq j}^{p} \hat{P}_s(\omega_{\ell(j)} \mid x \in \omega_{\ell(j),\ell(j'')})}{\sum_{j'=1}^{p} \prod_{j''=1, j'' \neq j'}^{p} \hat{P}_s(\omega_{\ell(j')} \mid x \in \omega_{\ell(j'),\ell(j'')})} \times \left(1 - \sum_{j'=p+1}^{c} \hat{P}_f(\omega_{\ell(j')} \mid x)\right), \qquad (11)$$

where the first term is related to the second stage, while the second term is related to the first stage. The objective of this second term is to maintain the sum of all the probabilities equal to one.

3.3. Control of the accuracy-speed tradeoff

Finally, to classify a pattern, we use a dynamic number $p(p\text{-}1)/2$ of SVCs. Hence, the smaller the threshold ε is, the larger the number p will tend to be. Indeed, if ε is too large, then we will never use the second stage of classification. But, if ε is too small, then the system uses unnecessary SVCs. Thus, this parameter controls the tolerance level of the first stage of classification and consequently the classifying cost. To fix it, we can observe on the validation dataset the tradeoff before error rate and KEPP and choose this value according to the constraints fixed by the application.

As we can see in Fig.5, while using a threshold of 10^{-3}, it is possible to obtain the same error rate of 1.53% than with the full "pairwise coupling".

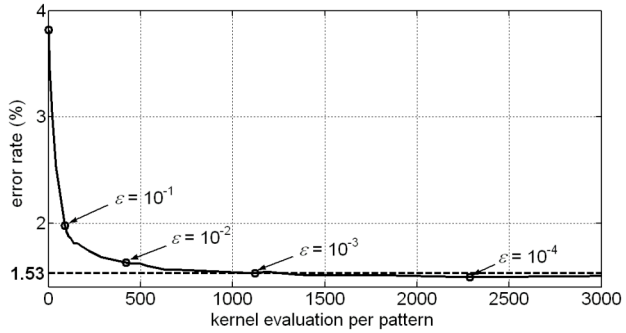

Figure 5. Accuracy-speed tradeoff on the validation dataset

Moreover, the use of smaller threshold ($\varepsilon = 10^{-4}$) allows a slightly better error-reject tradeoff (see Fig.6), but the number of KEPP is multiplied by two. For this reason, we fix the tolerance threshold ε at 10^{-3}, which seems a good compromise between accuracy and speed.

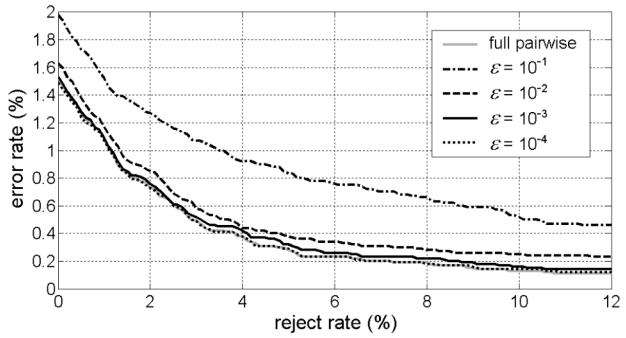

Figure 6. Error-reject tradeoff of our two-stage classification system on the validation dataset

Also, while the number p of SVCs used is dynamic, it is interesting to observe the distribution of p. Hence, we can see in Fig.7 that with our threshold of 10^{-3}, the half of the examples are processed without SVC, which confirms the previous remark related to Fig.4.

Finally, our two-stage system uses a mean of 1,120.1 KEPP and obtained on the test dataset an error rate of 1.50 %, which is comparable to the result of the full "pairwise coupling" (1.48 %).

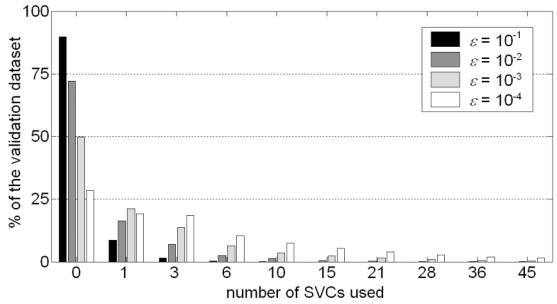

Figure 7. Distribution of the number *p* of SVCs used to classify the validation dataset

4. Conclusions and perspectives

With the objective to speeding up the decision making of Support Vector Classifiers, in the context of multi-class classification, we proposed a two-stage system embedded within a probabilistic framework. The results on the MNIST database show that the use of the first stage to pre-estimate probabilities allows to reduce the classifying cost by a factor 7.7, while preserving the accuracy of the full "pairwise coupling". Indeed, if we express the computational complexity in number of floating point operations (FLOPs), a kernel evaluation requires 2,355 FLOPs and a projection distance evaluation requires 81,510 FLOPs. Thus, the computational cost necessary to classify a pattern is approximately 32.4 MFLOPs with the "one against all" strategy, 26.2 MFLOPs with the "pairwise coupling" strategy, only 0.8 MFLOPs with the model-based approach and an average of 3.4 MFLOPs with our dynamic two-stage process.

On the other hand, although the "one against all" strategy estimates probabilities slightly better than the "pairwise coupling" strategy, it seems less adapted to large multi-class problems. Indeed, this approach is less modular than the "pairwise coupling" and consequently, the training is much slower and the test speedup related to the use of a two-stage architecture is less effective.

Moreover, while this implementation is only a proof of concept, several aspects can be improved in future works. Indeed, the model-based approach used in the first stage is not accurate. Thus, the use of a mixture of hyperplanes to model each class instead of one single hyperplane per class should improve significantly the accuracy of the first stage and consequently to speedup the decision making.

Table 1. Results of the different approaches on the test dataset

error rate (%)		0.5	0.4	0.3	0.2	0.1
reject rate (%)	**model-based approach**	12.68	13.74	16.97	20.01	28.59
	our two-stage system	3.31	3.99	4.94	6.57	9.85
	"pairwise coupling"	3.29	4.00	5.13	6.34	9.55
	"one against all"	2.41	2.92	3.60	4.42	6.32

Furthermore, our method is complementary to the method proposed in [2], which reduces the complexity of each SVC, while our approach reduces the complexity of the ensemble. Hence, the combination of the two approaches would allow fast classification. In addition, to improve the generalization performance, it is possible to incorporate known invariance of the problem, as in [2] where the authors generate artificial training examples, or as in [8] where discriminative features are extracted and allows to reduce the error-rate to only 0.4 %.

Finally, the probability estimation supposes the absence of outliers or more exactly a negligible prior probability of the corresponding class ω_0. But, for certain applications, it can be preferable to detect this type of noise. In this context, the use of a model-based approach at the first stage is another advantage, because as it is shown in the literature, this type of approach is able to reject efficiently the outliers.

5. References

[1] Bellili, A., Gilloux, M., Gallinari, P. (2003) An MLP-SVM combination architecture for offline handwritten digit recognition, *International Journal on Document Analysis and Recognition*, 244-252.

[2] Burges, C.J.C., Scholkopf, B. (1997) Improving the Accuracy and Speed of Support Vector Machines, *Advances in Neural Information Processing Systems*, 375-381.

[3] Chang, C.-C., Lin, C.-J. (2001) *LIBSVM : a library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

[4] Fumera, G., Roli, F., Giacinto, G. (2000) Reject option with multiple thresholds, *Pattern Recognition*, 2099-2101.

[5] Hamamura, T., Mizutani, H., Irie, B. (2003) A multiclass classification method based on multiple pairwise classifiers. *International Conference on Document Analysis and Recognition*, 809-813.

[6] Kimura, F., Takashina, K., Tsuruoka, S., Miyake, Y. (1987) Modified Quadratic Discriminant Functions and the Application to Chinese Character Recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 149-152.

[7] Lin, H.-T., Lin, C.-J., Weng, R.C. (2003) *A note on Platt's probabilistic outputs for support vector machines*. Technical report, Department of computer science and information engineering, National Taiwan University. Available at http://www.csie.ntu.edu.tw/~cjlin/papers.html

[8] Liu, C.-L., Sako, H., Fujisawa, H. (2003) Handwritten digit recognition: benchmarking of state-of-the-art techniques, *Pattern Recognition*, 2271-2285.

[9] Oh, I.-S., Suen, C.Y. (2002) A class-modular feedforward neural network for handwriting recognition. *Pattern Recognition*, 229-244.

[10] Parrado-Hernandez, E., Mora-Jimenez, I., Arenas-Garcia, J., Figuera-Vidal, A.-R., Navia-Vasquez, A. (2003) Growing support vector classifiers with controlled complexity, *Pattern Recognition*, 1479-1488.

[11] Platt, J. C. (1999) Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, *Advances in Large Margin Classifiers*, 61-74.

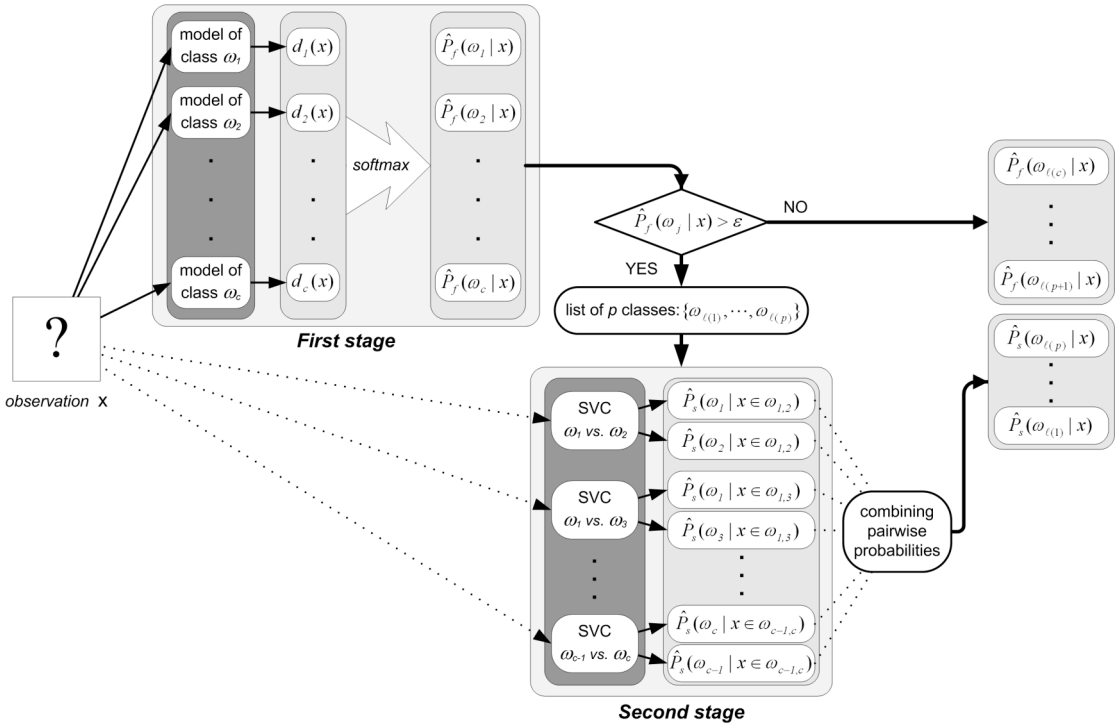

Figure 8. Overview of our two-stage classification system