A Compromise between Accuracy and Stability for RBF Kernel Selection in Relevance Vector Machines for Classification

Dmitry A. Kropotov, Dmitry P. Vetrov

Dorodnicyn Computing Centre of Russian Academy of Sciences,

Vavilova str. 40, Moscow, 119991, Russian Federation

e-mail: dkropotov@yandex.ru, vetrovd@yandex.ru

tel. +7 095 1354498, fax. +7 095 1356159

Nikita O. Ptashko

Lomonosov Moscow State University,

Vorob'evy gory, Moscow, 119234, Russian Federation

e-mail: ptashko@inbox.ru

tel. +7 095 9394202

Abstract

The task of RBF kernel selection in relevance vector machines (RVM) classifier is considered. RVM exploits a probabilistic Bayesian learning framework offering number of advantages to state-of-the-art support vector machines. In particular RVM effectively avoids determination of regularization coefficient C via evidence maximization. In the paper we show that RBF kernel selection in Bayesian framework requires extension of classifiers model. In new model integration over posterior probability becomes computationally unavailable. Therefore point estimation of posterior probability is used. In RVM evidence value is calculated via Laplace approximation. However extended model doesn't allow maximization of posterior probability as dimension of optimization parameters space becomes too high. Hence Laplace approximation can be no more used in new model. We propose a method of local evidence estimation which establishes a compromise between accuracy and stability of classifier. In the paper we first briefly describe maximal evidence principle, present model of kernel classifiers as well as our approximations for evidence estimation, and then give results of experimental evaluation.

Keywords

Machine Learning; Statistical Pattern Recognition; Kernel Selection; Stability of Classifiers; Bayesian Regularization; Relevance Vector Machines.

I. Introduction

In classification problem we are given a set of m input objects $\{\vec{x}_i\}_{i=1}^m$ and corresponding class labels $\{t_i\}_{i=1}^m$. Using this training information we would like to learn a model of algorithms, which can accurately determine class labels for new (previously unseen) input objects. One of the most popular algorithms for solving classification problems is Support Vector Machines (SVM) [1]. This algorithm makes prediction based on the following model:

$$y(x) = \sum_{i=1}^{m} w_i K(x, x_i) + w_0 \tag{1}$$

where $\{w_i\}_{i=0}^m$ is a set of real variables, kernel 'weights' and $K(\cdot, \cdot)$ is a kernel function.

SVM has proved its good performance on numerous tasks. The main reasons for the success of SVM are the following. Vapnik's idea of optimal hyperplane construction lead to maximal margin principle [2] which provides better generalization ability. Then so-called "kernel trick" allows application of linear methods of machine learning for constructing non-linear surfaces. However, successful application of SVM needs choosing the particular kernel function as well as regularization coefficient C. Different values of C and forms of kernel functions lead to sufficiently different behaviour of SVM for particular task.

Usually the parameters of kernel function and coefficient C are defined using a cross-validation procedure. This may be too computationally expensive. Moreover the cross-validation estimates of performance, although unbiased [2], have large variance due to the limited size of the learning sample.

The model selection problem for SVM awakens keen interest from number of researchers. Thus Vapnik and his colleagues suppose to perform kernel selection via theoretical bounds on leave-one-out test error. So-called R^2W^2 criterion function [9] and bound based on more subtle notion of *span of support vectors* [10] were elaborated. In [6] test error estimated with validation set is used. The work [7] applies evolutionary strategy for tuning parameters obtained by grid search procedure, which is time consuming. The popular way is application of Bayesian learning framework and MacKay's

maximal evidence principle [3] for model selection in SVM. Usually some probabilistic interpretation of SVM is provided which is then used for adaptation of maximal evidence principle [11], [8].

Recently Tipping proposed an algorithm, which also makes prediction based on decision rule (1). The algorithm exploits Bayesian regularization for best weights selection [4]. It was called Relevance Vector Machines (RVM). In this algorithm the weights of the so-called relevance vectors are interpreted as random values with gaussian prior distribution centered at zero. This approach doesn't require setting of regularization coefficient C for restriction of weights' values as large weights are penalized automatically during training. However, the problem of kernel selection still remains. In the paper we propose a method for kernel selection in RVM. We focus on the most popular RBF kernel functions and selection of parameter σ - width of Gaussian. We show that application of Bayesian framework for kernel selection requires extension of classifiers model. The extended model includes kernel centres. Integration over posterior probability in new model becomes computationally unavailable. That is why we use point estimate of posterior probability. Laplace approximation of evidence requires maximization of posterior probability as well as its Hessian computation. However, in new model too high dimension of optimization parameters space and the fact that posterior probability is multi-modal function makes application of Laplace approximation impossible. Instead of this we propose a method of local evidence estimation which lead to a compromise between stability and accuracy of classifier.

The article is organized as follows. Sections 2 and 3 briefly summarize ideas of Bayesian learning, maximal evidence principle and relevance vector machines classification. Section 4 presents our approach and gives points why maximal evidence principle can't be used directly for kernel selection. In section 5 experimental results on toy problems and real data are provided while the last section gives conclusion and discussion.

II. Bayesian Learning and Maximal Evidence Principle

The paradigm of Bayesian learning allows choosing the most appropriate model for the given training data. The term model in this context means a set of classifiers with fixed number of parameters and their prior distributions. Suppose that we have a set of models (either finite, countable or continuum) $W(\vec{\alpha}),\ \vec{\alpha} \in A$. Here $\vec{\alpha}$ defines the family of classifiers, the structure of their parameters $\vec{w}$, and their prior distributions $P(\vec{w}|\vec{\alpha})$. Denote by $P(D_{train}|\vec{w})$ the likelihood of the training data description with given values of $\vec{w}$. As the hyperparameters $\vec{\alpha}$ do not have direct influence on the training data we may write

$$P(D_{train}|\vec{w},\vec{\alpha}) = P(D_{train}|\vec{w}) \tag{2}$$

This means that $\vec{\alpha}$ affects the likelihood of the training data description only by means of its influence on $\vec{w}$. A classical way of classifier training is based on maximal likelihood principle, that is finding

$$\vec{w_{ML}} = \arg\max_{\vec{w}} P(D_{train}|\vec{w})$$

The probability of new data D_{test} given the training set is then just

$$P(D_{test}|D_{train}) = P(D_{test}|\vec{w_{ML}})$$

An alternative way of classifier training is to use Bayesian estimation of the posterior probability of $\vec{w}$

$$P(\vec{w}|D_{train}) = \frac{P(D_{train}|\vec{w})P(\vec{w})}{\int\limits_W P(D_{train}|\vec{w})P(\vec{w})d\vec{w}}$$

Then

$$P(D_{test}|D_{train}) = \int\limits_W P(D_{test}|\vec{w})P(\vec{w}|D_{train})d\vec{w}$$

Such inference can be done within one model. Now suppose we have several (or even continuum) models $W(\vec{\alpha})$ of different nature, complexity etc. The question is which model is preferable. To answer it we should estimate the so-called evidence

$$P(D_{train}|\vec{\alpha}) = \int\limits_{W(\vec{\alpha})} P(D_{train}|\vec{w})P(\vec{w}|\vec{\alpha})d\vec{w} \tag{3}$$

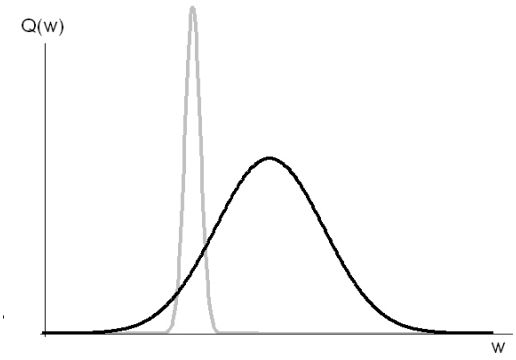

Fig. 1. The values of $Q(\vec{w})$ with respect to different $\vec{w}$ for two models. Note that although there is very good algorithm (sharp peak) in the first model (grey line), the second model (black line) has larger evidence (square under the curve) and hence is more preferable due to the larger rate of "good" algorithms (with large values of $Q(\vec{w})$).

Hereinafter for simplicity reasons we sometimes refer to $P(D_{train}|\vec{w})P(\vec{w}|\vec{\alpha})$ as $Q_{\vec{\alpha}}(\vec{w})$. The known principle of maximal evidence [3] states that we should choose that model which has the greatest value of evidence or, in other words, where the rate of "good" classifiers is the largest (see Fig. 1). Taking into account (2) the likelihood of the test data is calculated in the following way:

$$P(D_{test}|D_{train}) = \int\limits_{A}\int\limits_{W(\vec{\alpha})} P(D_{test}|\vec{w},\vec{\alpha})P(\vec{w},\vec{\alpha}|D_{train})d\vec{w}d\vec{\alpha} = \tag{4}$$

$$\int\limits_{A}\int\limits_{W(\vec{\alpha})} P(D_{test}|\vec{w})P(\vec{w}|\vec{\alpha},D_{train})P(\vec{\alpha}|D_{train})d\vec{w}d\vec{\alpha},$$

where

$$P(\vec{\alpha}|D_{train}) \propto P(D_{train}|\vec{\alpha})P(\vec{\alpha}),$$

i.e. in case of absence of any prior assumptions on $\vec{\alpha}$, $P(\vec{\alpha}|D_{train})$ is proportional to evidence. Integration over A is often intractable that is why $P(\vec{\alpha}|D_{train})$ is usually approximated by $\delta(\vec{\alpha_{MP}})$ where $\vec{\alpha_{MP}} = \arg\max\limits_{\vec{\alpha}} P(D_{train}|\vec{\alpha})$. Then equation (4) turns into

$$P(D_{test}|D_{train}) \approx \int\limits_{W(\vec{\alpha_{MP}})} P(D_{test}|\vec{w})P(\vec{w}|\vec{\alpha_{MP}},D_{train})d\vec{w} \tag{5}$$

III. Relevance Vector Machines

Here we briefly consider the idea proposed by Tipping on using Bayesian framework in kernel methods [4]. Henceforth we consider the classification problem. Let $D_{train} = \{\vec{x}, \vec{t}\} = \{x_i, t_i\}_{i=1}^m$ be training sample where $x_i = (x_i^1, \ldots, x_i^n)$ are feature vectors in an n-dimensional real space and t_i are class labels taking values from $\{-1, 1\}$. Consider the family of classifiers $y(x_{new}) = \text{sign}(\sum_{i=1}^m w_i K(x_{new}, x_i) + w_0) = \text{sign}(h(x_{new}, \vec{w}))$. Establish prior distribution on the weights $P(w_i|\alpha_i) \sim N(0, \alpha_i^{-1})$. The set of parameters $\vec{\alpha}$ determines the model in which the posterior distribution is looked for. Define the likelihood of training sample as

$$P(D_{train}|\vec{w}, \vec{\alpha}) = P(D_{train}|\vec{w}) = \prod_{i=1}^{m} \frac{1}{1 + \exp(-t_i h(x_i, \vec{w}))}$$

Then the evidence of model is given by (3). Our goal is to find $\vec{\alpha}$ which maximizes evidence and then to get posterior distribution $P(\vec{w}|D_{train}, \vec{\alpha})$. As direct calculation of (3) is impossible due to the intractable integral, Tipping used Laplace approximation for its estimation. He approximated $L_{\vec{\alpha}}(\vec{w}) = \log(P(D_{train}|\vec{w})P(\vec{w}|\vec{\alpha}))$ by quadratic function using its Taylor decomposition with respect to $\vec{w}$ at the point of maximum $\vec{w_{MP}}$. Such approximation can be integrated yielding

$$P(D_{train}|\vec{\alpha}) \approx Q_{\vec{\alpha}}(\vec{w_{MP}}) \mid \Sigma \mid^{1/2}, \tag{6}$$

$$\Sigma = (-\nabla_{\vec{w}} \nabla_{\vec{w}} L_{\vec{\alpha}}(\vec{w}) \mid_{\vec{w}=\vec{w_{MP}}})^{-1} = (-\nabla_{\vec{w}} \nabla_{\vec{w}} \log(P(D_{train} \mid \vec{w})) - A)^{-1} \tag{7}$$

where $A = diag(\alpha_1, \ldots, \alpha_m)$. Differentiating the last expression with respect to $\vec{\alpha}$ and setting the derivatives to zero gives the following iterative re-estimation equation

$$\alpha_i^{new} = \frac{\gamma_i}{w_{MP,i}^2} \tag{8}$$

$$\gamma_i = 1 - \alpha_i^{old} \Sigma_{ii} \tag{9}$$

Here γ_i is so-called effective weight of i^{th} parameter. It shows how much it is constrained by regularization term established by prior. It can be easily shown that $\gamma_i \in [0, 1]$. If α_i is close to zero, w_i is almost unconstrained and γ_i is close to one. On the contrary in case of large α_i the corresponding

parameter w_i is close to zero and is not much affected by training information. So its effective weight tends to zero.

The training procedure consists of three iterative steps. First we search for the maximum point $\vec{w_{MP}}$ of $L(\vec{w})$. Then we make approximation according to (6) and use (8) to get the new values of $\vec{\alpha}$. The steps are repeated until the process converges.

After the training is finished the integral (5) can be approximated by setting $P(\vec{w}|D_{train}, \vec{\alpha}) \approx \delta(\vec{w_{MP}})$ resulting in the expression

$$P(D_{test}|D_{train}) = P(D_{test}|\vec{w_{MP}}) \tag{10}$$

It was shown [4] that RVM provides approximately the same quality as SVM with the same kernel function and best value of C selected by cross-validation but does not require the regularization coefficient C to be set by user. Moreover it appeared that RVM is much more sparse, i.e. the rate of non-zero weights (relevance vectors) is significantly less than the rate of support vectors. This happens because most of the objects are treated as irrelevant and the corresponding α tend to infinity.

IV. Kernel Selection for RVM

Although maximal evidence principle is fully given in probabilistic terms we may suggest another interpretation for it. Equation (6) can be viewed as a compromise between accuracy of algorithm on the training sample (the value of $Q_{\vec{\alpha}}(\vec{w_{MP}})$) and its stability with respect to small changes of parameters (expressed by squared root of inverse Hessian determinant). Then we may formulate *stability principle*. The more "stable" the classifier is, the better is its generalization ability. The notion of stability is quite informal. Different definitions of stability and its relation to generalization ability were investigated by many researches [13], [14]. Here we understand stability as ability to keep large likelihood (or to be more exact the values of $Q_{\vec{\alpha}}(\vec{w})$) as long as possible moving from the point of maximum. Such view allows to modify the concept of Bayesian regularization for the cases where its direct application is impossible or not reasonable.

Model selection via maximal evidence principle allows avoiding the direct setting of weight constraints in RVM. Nevertheless, selection of appropriate kernel function is still needed. The question is whether it is possible to treat the kernel function type as meta-parameter and to use Bayesian approach for its definition. Henceforth we consider popular RBF parametric kernels $K(x,z) = \exp(-\frac{||x-z||^2}{2\sigma^2})$. Our goal is to find the best σ value without cross-validation using the idea of stability.

It is easy to see that small values of σ lead to overfitting and hence to high accuracy on the training sample. On the other hand second item of equation (6) does not penalize such σ due to the following reason. Small σ means that almost all objects from the training set have non-zero weights and the influence from the neighboring objects can be neglected. Variations of object's weight just change the height of the corresponding kernel function still keeping its center in the object. The change of weight regardless its value affects only the object in the center of kernel. This means that such weight's modification cannot change $L_{\vec{\alpha}}(\vec{w})$ much. The likelihood after modification is still very high and the second term in (6) even encourages small σ as the algorithm is more stable with respect to the changes of weights in this case. At the same time if we start moving the position of the kernel center, the likelihood of the training object changes dramatically (see fig.2). So small σ makes classification unstable with respect to shifts of the kernel centers.

Actually the stability with respect to weight changes is important when we select regularization coefficients $\vec{\alpha}$. The parameters of kernel functions are responsible for stability with respect to kernel shifts. Hence kernel selection requires extension of our model (1) to $h_E(x_{new}, \vec{w}, \vec{z}) = \text{sign}(\sum_{i=1}^{m} w_i K(x_{new}, z_i) + w_0)$. In new model direct calculation of evidence (3) remains impossible due to intractable integral. Application of Laplace approximation for evidence requires optimization of kernels location z_i in order to maximize

$$L_{\sigma,\vec{\alpha}}(\vec{w}, \vec{z}) = log(P(D_{train}|\vec{w}, \vec{z})P(\vec{w}|\vec{\alpha})P(\vec{z})) \tag{11}$$

Unfortunately optimization of $L_{\sigma,\vec{\alpha}}(\vec{w}, \vec{z})$ with respect to $\vec{z}$ is too difficult due to large amount of dimensions as $\vec{z} \in R^{mn}$. Moreover unlike the case with weights, the expression (11) is non-linear with

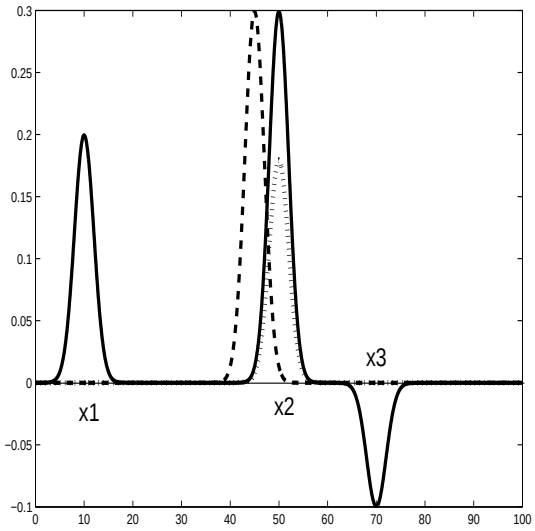

Fig. 2. The likelihood of the training sample is a product of likelihoods in each training object $x1, x2, x3$. Narrow Gaussians centered in training objects have nearly no influence on the other objects from the training set. Small weight change still keeps the likelihood of the corresponding object high enough (dotted line) while small shifts of a relevant point (gaussian center) make likelihood significantly lower in case of small σ (dashed line).

respect to kernel centers and hence multi-modal. This hardens optimization even more.

Consider the last point a bit closer. According to maximal evidence principle model quality depends on the value of integral (3). Decision rule should be constructed with the aid of equation (5). But in our case this is impossible due to intractable integral. Hence we would prefer using only the classifier which was obtained via maximization of $L_{\sigma,\vec{\alpha}}(\vec{w},\vec{z})$. If function $L_{\sigma,\vec{\alpha}}(\vec{w},\vec{z})$ were unimodal then the evidence could be approximated by its local behaviour at the maximum point $(\vec{w_{MP}},\vec{z_{MP}})$. Now consider the following situation. Our solution is located in narrow peak at point $(\vec{w_{MP}},\vec{z_{MP}})$ but there is good stable algorithm somewhere else within the model. The evidence of obtained answer will be high, but the generalization ability of this single classifier is still poor (see fig. 3). Of course if we used integration (5) for decision-making that would do, but the trouble is we have no choice but to use its point estimate (10). According to stability principle we should consider only local characteristics of the point which will be taken as final solution. Such characteristics are the value of function $L_{\sigma,\vec{\alpha}}(\vec{w},\vec{z})$ and its derivatives which represent the measure of instability. The analogies with Bayesian framework can be used to unite these values in one equation.

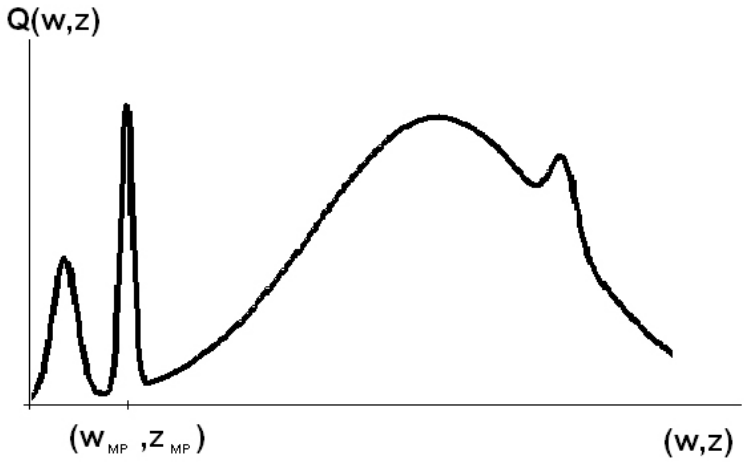

Fig. 3. The example of model which has large value of evidence with quite poor point estimate. There is no profit of large evidence value if we use algorithm with $(\vec{w},\vec{z}) = (w_{\vec{MP}}, z_{\vec{MP}})$. At the same time local characteristics of point $(w_{\vec{MP}}, z_{\vec{MP}})$ such as $\nabla_{w,z}\nabla_{w,z}Q(\vec{w},\vec{z})\mid_{(\vec{w},\vec{z})=(w_{\vec{MP}},z_{\vec{MP}})}$ penalize the obtained algorithm belonging to the model.

As it was mentioned above the optimization of kernel locations is very difficult and time-consuming. Moreover, according to our experiments such optimization gives nearly no profit in accuracy while the time of training increases significantly. That is why we propose to keep training objects as kernel centers estimating at the same time algorithm's stability with respect to hypothetical kernel shifts. Consequently it often happens that $\frac{\partial L_{\sigma,\vec{\alpha}}(\vec{w},\vec{z})}{\partial z_i}\mid_{z_i=x_i} \neq 0$. Hence both first and second derivatives become necessary for definition of stability measure. Our goal is to develop some kind of generalization of expression (6) for the multi-modal case and non-extremum point.

We will estimate stability of already trained classifier with respect to kernel shifts. Then we may treat $w_{\vec{MP}}$ as constants which do not depend on $\vec{z}$. It seems quite obvious that there are no prior constrains on centers location so that we may establish improper uniform prior $P(\vec{z}) = const$. Now we may rewrite the expression (11):

$$L_{\sigma,\vec{\alpha}}(\vec{w},\vec{z}) = \log(P(D_{train}|\vec{w},\vec{z})) + \Psi(\vec{w},\vec{\alpha}) \tag{12}$$

where $\Psi(\vec{w},\vec{\alpha})$ does not depend on $\vec{z}$ and hence may be ignored during differentiation.

When the training is finished all we are interested in is to keep accuracy on the test sample as

close to the training accuracy as possible. We are not going to get better performance (although it is theoretically possible). This means that we should examine how fast the accuracy may degrade with respect to the changes of algorithm's settings (in our case these are kernel centers). Denote A_i the instability of $P(D_{train}|w_{MP}^{\vec{}}, \vec{z})$ with respect to kernel located in z_i. We assume that it may be decomposed as if the stabilities with respect to different coordinates were independent

$$A_i = \prod_{j=1}^{n} A_{ij}$$

where A_{ij} expresses the stability of classifier with respect to small changes of j^{th} coordinate of i^{th} kernel. For determination of A_{ij} we will approximate $\log(P(D_{train}|w_{MP}^{\vec{}}, \vec{z}))$ with parabolic function using its Taylor decomposition at the point $\vec{z} = \vec{x}$ with respect to z_i^j. Then we may write

$$A_{ij} = \begin{cases} |a|^{-1}, & if\ b \geq 0 \\ \frac{1}{2}\sqrt{\frac{2\pi}{b}} \exp\left(\frac{a^2}{2b}\right)\left(1 - erf\left(\frac{|a|}{\sqrt{2b}}\right)\right), & otherwise \end{cases} \tag{13}$$

here

$$a = \frac{\partial \log(P(D_{train}|w_{MP}^{\vec{}}, \vec{z}))}{\partial z_i^j}$$

$$b = -\frac{\partial^2 \log(P(D_{train}|w_{MP}^{\vec{}}, \vec{z}))}{(\partial z_i^j)^2}$$

The sense of equation (13) is shown on figure IV. First we approximate $f(z_i^j) = P(D_{train}|w_{MP}^{\vec{}}, \vec{z})$ with negative parabola or with a line (if second derivative is non-negative) in logarithmic scale at point $z_i^j = x_i^j$. This yields to gaussian or exponent approximation of tail of $Q(\vec{w}, \vec{z})$. Then we integrate the tail of approximation in order to get a measure of stability in terms of derivatives. Note that we do not want to approximate the real value of integral. Our purpose is to express a degree of stability similar to (6). If x_i^j were an extremum point of $f(z_i^j)$ then A_{ij} would be exactly the result of Laplace approximation taken along x_i^j coordinate. Regarding evidence as a compromise between accuracy and stability rather then integral (3) we deduced similar expression for more general case.

Now we would like to unite the stability with accuracy in one expression. To do so we should consider the weight of each kernel. Actually if the weight of kernel is close to zero its stability doesn't play

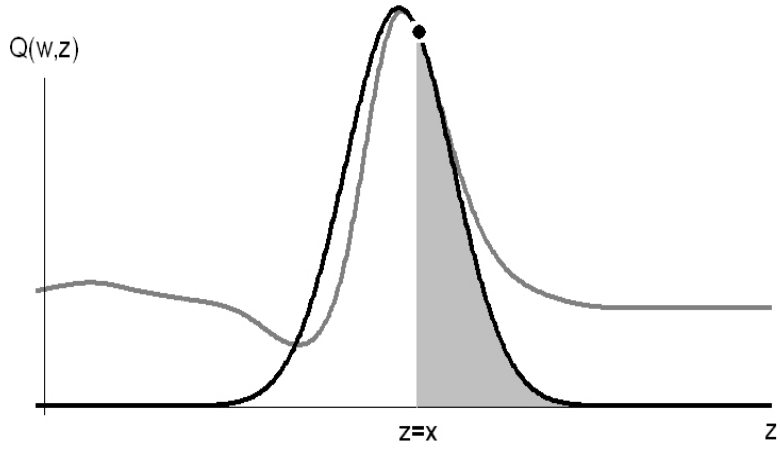

Fig. 4. The optimization in the space of kernel locations $\vec{z}$ is quite hard so we place all kernels in the objects of the training sample, i.e. $\vec{z} = \vec{x}$. This may lead to non-zero gradients of $\nabla_{\vec{z}} Q(\vec{w}, \vec{z})$. To establish a compromise between the accuracy of given algorithm and its ability to keep its performance with changes of parameters we integrate the tail in Laplace approximation of $Q(\vec{w}, \vec{z})$ (grey area). This yields to expression which combines algorithm's local accuracy (the value of $Q(\vec{w}, \vec{z})$) and stability (the values of its derivatives).

important role. It is clear that zero weight w_i corresponds to the case as if there were no kernel in z_i. Taking into consideration the effective weights (9) of each kernel γ_i which vary from 0 to 1 we may get the expression for total stability of likelihood with respect to all kernels

$$Z = \prod_{i=1}^{m} A_i^{\gamma_i} = \prod_{i=1}^{m} (\prod_{j=1}^{n} A_{ij})^{\gamma_i} \tag{14}$$

The last expression is equivalent to simple weighted summation of stabilities in logarithmic scale.

Theorem 1. The expression (14) is correctly defined with respect to addition or removal of kernels with zero weight.

Proof. Suppose that $w_1 = 0$. Then we should prove that

$$\prod_{i=1}^{m} A_i^{\gamma_i} = \prod_{i=2}^{m} A_i^{\gamma_i}$$

During RVM training we maximize evidence (6). If $w_1 = 0$ then it can be shown that corresponding α_1 equals infinity. Otherwise w_1 would fit the data at least partly. Now recall that $\gamma_1 = 1 - \alpha_1 \Sigma_{11}$ where Σ_{11} is the first element of inverse Hessian matrix (7)

$$\Sigma = (-\nabla_{\vec{w}} \nabla_{\vec{w}} L_{\vec{\alpha}}(\vec{w}) \mid_{\vec{w}=w_{\vec{MP}}})^{-1}$$

For large α_1 the element $\Sigma_{11} \approx \alpha_1^{-1}$. Thus the limit $\lim_{\alpha_1 \to \infty} \alpha_1 \Sigma_{11} = 1$. This yields to $\gamma_1 = 0$. Substituting this result in (14) we have

$$\prod_{i=1}^{m} A_i^{\gamma_i} = A_1^{\gamma_1} \prod_{i=2}^{m} A_i^{\gamma_i} = \prod_{i=2}^{m} A_i^{\gamma_i}$$

This means that we may add or remove arbitrary number of zero-weight kernels without changing anything in stability expression. Theorem is proved.

Multiplying Z and the value of likelihood at the point $w_{MP}^{\vec{}}$ we get *kernel validity* value

$$KV = P(D_{train}|w_{MP}^{\vec{}}, \vec{z})Z \tag{15}$$

Note that the equation (6) is just a particular case of kernel validity. The kernel function which corresponds to the largest value of validity is supposed to be the best one for the particular task.

Thus the procedure for selection of width parameter σ in gaussian parametric family of kernel functions becomes the following:

1. Choose some σ value.

2. Put $\vec{z} = \vec{x}$.

3. Run iterative process for training RVM classifier. At each step we first search for maximum point $w_{MP}^{\vec{}}$ of function $L_{\sigma,\vec{\alpha}}(\vec{w}, \vec{x})$. Then Laplace approximation (6) is used and new $\vec{\alpha}$ values are calculated by means of equations (8), (9). The steps are repeated until the process converges.

4. At the point $w_{MP}^{\vec{}}$ calculate kernel validity (15), where components A_{ij} are taken from (13), while effective weights γ_i are determined from (9).

The value of σ corresponding to the largest value of validity is considered to be the optimal one.

Our further task is to check performance of the proposed kernel selection procedure. It seems obvious that both very narrow and very wide kernels will have lower values of validity. Narrow kernels lead to too unstable classifiers with respect to kernel shifts while classifiers with wide kernels are too inaccurate. We expect to see such behavior during experiments on toy and real-world problems.

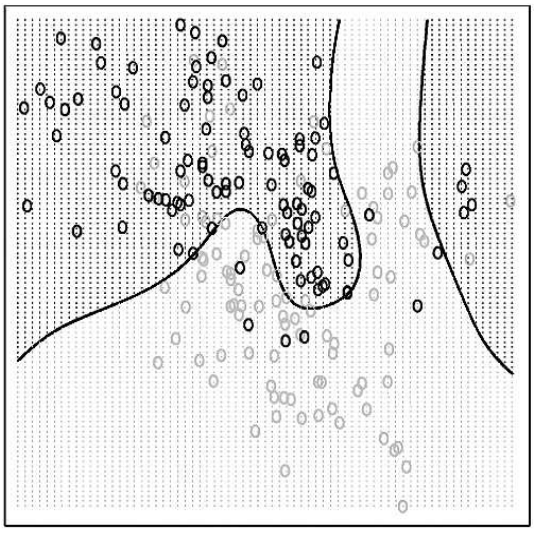

Fig. 5. Training sample of toy problem. Black line is optimal class boundary according to Bayes classifier.

V. Experimental Results

A. Toy problem

First we examined the performance on easy-to-interpret toy problem taken from [12]. It can be also downloaded from *http://www-stat.stanford.edu/ElemStatLearn.* This is two-class task with non-linear class border. The dimension of feature space is two. Training sample with optimal Bayes border between classes is given on figure 5. It consisted of 200 objects. As a test sample we generated 5000 objects according to the same distribution. The error rate on such large sample can be regarded as error on the universal set. The importance of selecting the proper value of kernel width is illustrated by figure 6 where train and test errors for different kernels are shown. To test the performance of the proposed approach we compared it with its most widely-used alternative - cross-validation. In particular we used 5-fold cross-validation. We also estimated validity of each kernel using the ideas described above. Figure 6 shows cross-validation error so as validity index. It can be seen that maximum validity kernel is better suited for the task (although not in the best way). The reason of relatively poor cross-validation performance is the following. In spite of its good property of being unbiased [2], cross-validation estimate may have large variance especially for small training samples. We suppose that this is also the reason that validity measure doesn't reach its maximum value on the

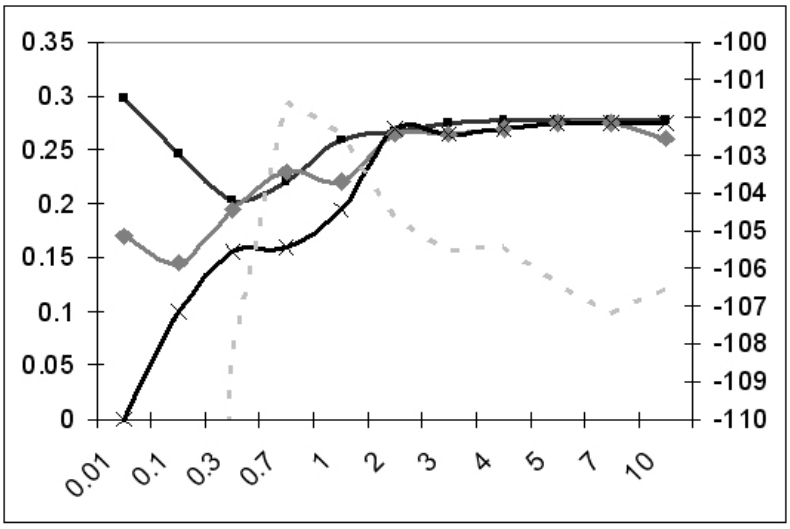

Fig. 6. Different measurements of kernel quality for toy problem. Black lines show train (cross markers) and test (square markers) errors correspondingly. Grey line with markers shows 5-fold cross-validation error. The logarithm of kernel validity is shown by light grey line. It can be seen that narrow kernels lead to overfitting while wide kernels can't achieve good performance even for the training sample. Note that neither cross-validation error, nor kernel validity index select best possible kernel with $\sigma = 0.3$. This can be explained by the fact that both measures are computed using only limited training sample. However the use of kernel validity index is preferable.

kernel which leads to the minimum of test error. Both cross-validation techniques and validity index are based on the training sample which, generally speaking, differs from the universal set due to its limited size.

B. Real world tasks

To compare performance of kernel validity index vs. cross-validation we carried out 180 experiments on kernel selection using 9 problems from UCI repository [5]. The experiments were conducted in the following way. For each task we randomly split the data into train (33%) and test (67%) sets. Then we trained and tested RVM using kernels of different width ($\sigma = 0.01, 0.1, 0.3, 1, 2, 3, 4, 5, 7, 10$), calculating cross-validation errors (using 5-fold cross-validation) and validity indices. After that we chose test error which corresponded to the kernels with maximum validity or with best cross-validation estimate and averaged these errors by 20 pairs of train/test tables formed from the same dataset. The averaged results with their standard deviations are shown in table I. Note that we also considered popular support vector machines (SVM) with the same kernels. Column RVM CV shows the averaged

results of kernel selection according to 5-fold cross-validation for RVM. The next column shows the analogous results obtained by applying 5-fold cross-validation to SVM. The following column RVM MV shows averaged test errors which correspond to maximum kernel validity index. Column SVM MV shows how SVM performs *with the same kernels* as in RVM MV. Note that we did not apply maximum validity procedure to select SVM kernel just taking the one which appeared to be "the best" for RVM case. Column SVM MV helps us to check whether the optimal kernel width is defined only by the problem itself or also by the training algorithm. Finally the last column contains minimal possible test error averaged by 20 pairs of train/test sets.

TABLE I

EXPERIMENTAL EVALUATION OF DIFFERENT KERNEL SELECTION METHODS FOR RVM AND SVM. THE TABLE PRESENTS TEST ERROR ALONG WITH STANDARD DEVIATION FOR DATASETS TAKEN FROM UCI REPOSITORY. COLUMNS RVM CV AND SVM CV STATE RESULTS FOR RVM AND SVM CORRESPONDINGLY WITH KERNEL FUNCTION OBTAINED VIA 5-FOLD CROSS-VALIDATION PROCEDURE. RVM MV CONTAINS RVM PERFORMANCE WITH MAXIMAL KERNEL VALIDITY INDEX. SVM MV SHOWS RESULTS OF SVM WITH THE SAME KERNEL AS IN RVM MV. MINTESTERROR PROVIDES MINIMALLY POSSIBLE TEST ERROR FOR EACH SAMPLE. THE ROW TOTAL PRESENTS TOTAL RATE OF EACH CLASSIFIER.

Sample Name	**RVM CV**	**SVM CV**	**RVM MV**	**SVM MV**	**MinTestError**
AUSTRALIAN	15.5 ± 1.2	16.5 ± 1.9	18.6 ± 4.35	21 ± 3.6	13.4
BUPA	41 ± 0.4	37.5 ± 2.5	39 ± 3.6	37.6 ± 3.8	31
CLEVELAND	18.6 ± 1.8	21 ± 2.7	20 ± 3.5	28 ± 5.6	17
CREDIT	17.3 ± 2.7	18 ± 1.6	16.9 ± 2.4	20 ± 2.9	14.5
HEPATITIS	43 ± 5.6	39.17 ± 3.8	39 ± 3.9	39.21 ± 4.6	36
HUNGARY	22 ± 4.4	20 ± 2.3	24 ± 5.3	26 ± 4	18
LONG BEACH	25.25 ± 0.5	25.18 ± 0.9	27 ± 4.7	26 ± 4.6	24.5
PIMA	34 ± 2.7	30 ± 2	27 ± 2.5	29.6 ± 2.9	23
SWITZERLAND	6.4 ± 1.6	8 ± 1.8	7 ± 2	7.6 ± 2.3	5.8
Total	**21**	**20**	**20**	**29**	

To interpret data from table I the results were rated in the following way. The least test error was given one point, while the second two points, etc. The worst result was assigned four points. Then

these points were summed by all nine problems from UCI repository. Total results are shown in last line of the table. According to it we can make several conclusions. First of all we may say that RVM and SVM show approximately the same competitive results although training algorithms are completely different so as the location of significant kernels. Our experiments confirmed that RVM is generally much more sparse than SVM generating 5-8 times less kernels than the corresponding support vector classifier. Another important conclusion is that our kernel validity measure works no worse than cross-validation alternative. Moreover it requires only one cycle of training and hence works several times faster. Very interesting effect is poor quality of SVM performance using the kernels which were considered to be the best (in the sense of our validity measure) for RVM. This proves that kernel type depends much on the method of training vector machine classifier. Also we should mention that neither cross-validation nor maximum validity index lead to minimum possible test error. This can be connected both with peculiarities of training sample and with the fact that test sample may be biased with respect to the universal set.

VI. Discussion and Conclusion

Results of experiments allow making the following conclusions. First of all the idea of stability can be used for generalization of maximal evidence principle. Unlike structural risk minimization which restricts too flexible classifiers and minimum description length approach which penalizes algorithmic complexity, the concept of Bayesian regularization (and its modification described above) tries to establish the model where the solution is stable with respect to changes of classifier parameters. Numerous realizations of Bayesian principles for model selection so as our experiments confirm that such approach to machine learning is promising. We decided to move from probabilistic approach and concentrate directly on idea of stability rather than on applying maximal likelihood principle to models (i.e. estimating evidence). Such modification allows exploiting this framework for non-linear models using only one classifier instead of integration over the whole space of parameters with weights

defined by posterior distribution $P(\vec{w}|D_{train}, \vec{\alpha})$. The proposed characteristic of kernel validity does not show how good the kernel is for particular task. It only can serve for estimation of kernel utility in case of fixed training procedure (in our case this is RVM). Performance of logistic regression or SVM classifier with the same kernel may differ significantly. This happens because we do not estimate the validity of whole model (as we use only one classifier with $\vec{w} = \vec{w_{MP}}$) but consider only local stability of $Q(\vec{w})$ at point $\vec{w_{MP}}$. This method seems to be quite general and probably could be applied to other complex machine learning algorithms for tuning their model parameters.

The idea to take into consideration both the model of algorithms and particular training procedure (our ability to find good classifier inside the model) for estimation of classifier's quality is not novel. For example, Vapnik proposes so-called effective capacity [2]. Unlike model capacity new notion suggests consideration of training sample and considers only those classifiers which can be obtained inside the model using particular training sample. As a result error bounds become more accurate. Popular boosting and bagging techniques are said to increase both training accuracy and generalization ability of algorithms. These methods make classifier's model sufficiently more complex. Nevertheless, effective way of choosing particular algorithm inside the extended model avoids drawbacks of such complication. Explicit consideration of training procedure together with model's properties led to new theory of algorithms quality estimates, based on combinatorial approach [15]. In our case we are not able to consider all possible algorithms inside the model (to integrate over posterior probability $P(\vec{w}|D_{train}, \vec{\alpha})$). However, consideration of local stability of $Q(\vec{w}, \vec{z})$ at point $\vec{w_{MP}}$ (our ability to find good classifier inside the model) gives us appropriate technique for kernel selection task.

In kernel selection procedure stability principle can be used for searching more sophisticated kernels (e.g. $K(\vec{x}, \vec{y}) = \exp(-\frac{1}{2}(\vec{x} - \vec{y})^T S^{-1}(\vec{x} - \vec{y}))$ where S is positively defined matrix) without risk of overfitting. The same ideas can be used in regression problems e.g. in relevance vector regression. It is very interesting how the principle of stability will change in application for support vector machines. Although SVM exploits non-probabilistic framework, it can be expressed in terms of likelihood and

prior [11].

Acknowledgements

This work was supported by the Russian Foundation for Basic Research (projects No. 05-07-90333, 05-01-00332, 04-01-00161, 04-01-08045) and INTAS (YS 04-83-2942).

References

[1] Burges, C.J.S: A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery **2** (1998) 121–167

[2] Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag New York (1995)

[3] MacKay, D.J.C.: Information Theory, Inference, and Learning Algorithms. Cambridge University Press (2003)

[4] Tipping, M.E.: Sparse Bayesian Learning and the Relevance Vector Machines. Journal of Machine Learning Research **1** (2001) 211–244

[5] Murphy, P.M., Aha, D.W.: UCI Repository of Machine Learning Databases [Machine Readable Data Repository]. Univ. of California, Dept. of Information and Computer Science, Irvine, Calif. (1996)

[6] Ayat, N.E., Cheriet, M., Suen, C.Y.: Optimization of SVM Kernels using an Empirical Error Minimization Scheme. Proc. of the First International Workshop on Pattern Recognition with Support Vector Machines (2002)

[7] Friedrichs, F., Igel, C.: Evolutionary Tuning of Multiple SVM Parameters. Neurocomputing **64** (2005) 107–117

[8] Gold, C., Sollich, P.: Model Selection for Support Vector Machine Classification. Neurocomputing, **55(1-2)** (2003) 221–249

[9] Weston, J., Mukherjee, S., Chapelle, O., Pontil, M., Poggio, T., Vapnik, V.: Feature Selection for Support Vector Machines. Proc. of 15th International Conference on Pattern Recognition, Vol.2 (2000)

[10] Chapelle, O., Vapnik, V.: Model Selection for Support Vector Machines. Advances in Neural Information Processing Systems 12, ed. S.A. Solla, T.K. Leen and K.-R. Muller, MIT Press (2000)

[11] Kwok, J.T.-Y.: The Evidence Framework Applied to Support Vector Machines. IEEE-NN, **11(5)** (2000)

[12] Friedman, J., Hastie, T., Tibshirani, R.: The Elements of Statistical Learning. Springer (2001)

[13] Kutin, S., Niyogi, P.: Almost-everywhere algorithmic stability and generalization error. Tech. Rep. TR-2002-03: University of Chicago (2002)

[14] Bousquet, O., Elisseeff, A.: Algorithmic stability and generalization performance. Advances in Neural Information Processing Systems 13 (2001)

[15] Vorontsov, K.V.: Combinatorial substantiation of learning algorithms. Journal of Comp. Maths Math. Phys. **44(11)** (2004) 1997-2009 *http://www.ccas.ru/frc/papers/voron04jvm-eng.pdf*