

A New Learning Formulation for Kernel Classifier Design

Atsushi Sato

Information and Media Processing Laboratories
NEC Corporation
Kawasaki, Japan

Abstract—This paper presents a new learning formulation for classifier design called “General Loss Minimization.” The formulation is based on Bayes decision theory which can handle various losses as well as prior probabilities. A learning method for RBF kernel classifiers is derived based on the formulation. Experimental results reveal that the classification accuracy by the proposed method is almost the same as or better than Support Vector Machine (SVM), while the number of obtained reference vectors by the proposed method is much less than that of support vectors by SVM.

Keywords—learning method; loss minimization; Bayes decision theory; kernel classifiers; support vector machine

I. INTRODUCTION

Prototype-based classifiers are simple but achieve good performance in real applications. The author proposed Generalized Learning Vector Quantization (GLVQ) [1], [2] for prototype learning, and its promising performance has been shown in many applications [3]. Meanwhile, kernel classifiers have been investigated in recent years. Many works have shown that Support Vector Machine (SVM) [4] outperforms conventional prototype-based classifiers. An extension of GLVQ by means of kernel functions has been also investigated [5], but its superiority to kernel classifiers such as SVM seems to be not clarified.

In this paper, a new learning formulation called “General Loss Minimization” is proposed for classifier design. The formulation is based on Bayes decision theory which can handle various losses as well as prior probabilities. A learning method for kernel classifiers is derived based on that. Experiments for two-dimensional artificial data and for the UCI machine learning database [6] were conducted to reveal the effectiveness of the proposed method to SVM.

II. GENERAL LOSS MINIMIZATION

The GLVQ algorithm is derived based on Minimum Classification Error (MCE) [7]. MCE is regarded as a general framework of classifier design, but it cannot deal with various losses except zero-one loss, because L_p -norm is used for the misclassification measure. In this section, a more general formulation of learning is proposed which can handle various losses and prior probabilities.

According to Bayes decision theory, an expected loss is

defined by

$$L = \sum_{k=1}^K \sum_{j=1}^K L_{kj} P(\omega_k) \int_{\mathcal{R}_j} p(\mathbf{x}|\omega_k) d\mathbf{x}, \quad (1)$$

where K denotes the number of classes, \mathbf{x} is a d -dimensional vector-valued random variable, L_{kj} is a loss for assigning \mathbf{x} arising from class ω_k to class ω_j , $P(\omega_k)$ is the prior probability of class ω_k , $p(\mathbf{x}|\omega_k)$ is the class-conditional probability density function for \mathbf{x} , and \mathcal{R}_j is a decision region for class ω_j . The meaning of Eq. (1) is as follows: if \mathbf{x} arising from class ω_k falls into the decision region of class ω_j , this probability is multiplied by the prior probability of class ω_k and by the loss L_{kj} , and these are summed up for every class ω_k of \mathbf{x} and for every class ω_j of decision regions. The classifier whose decision regions minimize the expected loss can be regarded as the best performing classifier according to Bayes decision theory.

In practice, the class-conditional probability density function $p(\mathbf{x}|\omega_k)$ is unknown, so it should be approximated by sample density as follows:

$$p(\mathbf{x}|\omega_k) \simeq \frac{1}{N_k} \sum_{n=1}^N \delta(\mathbf{x} - \mathbf{x}_n) 1(\mathbf{x}_n \in \omega_k), \quad (2)$$

where \mathbf{x}_n is the n -th input vector (or feature vector), N is the number of all input vectors, N_k is the number of input vectors which belongs to class ω_k (i.e., $\sum_k N_k = N$), and $\delta(\cdot)$ is the delta function. $1(\cdot)$ is an indicator function such that $1(\text{true})=1$ and $1(\text{false}) = 0$. By substituting Eq. (2) to Eq. (1), we can obtain the following equation called “empirical loss”:

$$L \simeq \sum_{n=1}^N \sum_{k=1}^K \sum_{j=1}^K \frac{L_{kj} P(\omega_k)}{N_k} 1(\mathbf{x}_n \in \mathcal{R}_j) 1(\mathbf{x}_n \in \omega_k). \quad (3)$$

In the above equation, only the decision regions are related to classifier parameters. Therefore, if the decision regions can be described explicitly by using the set of classifier parameters $\boldsymbol{\alpha}$, we can formulate the learning of classifiers as the minimization of the above equation. To do this, we rewrite the above equation as follows:

$$L(\boldsymbol{\alpha}) = \sum_{n=1}^N \sum_{k=1}^K \sum_{j=1}^K \frac{L_{kj} P(\omega_k)}{N_k} \ell(\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})) 1(\mathbf{x}_n \in \omega_k), \quad (4)$$

where $\ell(\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha}))$ is a sort of a risk associated with assigning \mathbf{x}_n to class ω_j . $\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})$ is the misclassification measure of \mathbf{x}_n defined by

$$\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha}) = \frac{g_j(\mathbf{x}_n; \boldsymbol{\alpha}) - g_k(\mathbf{x}_n; \boldsymbol{\alpha})}{g_j(\mathbf{x}_n; \boldsymbol{\alpha}) + g_k(\mathbf{x}_n; \boldsymbol{\alpha})}, \quad (5)$$

where $g_j(\cdot)$ and $g_k(\cdot)$ are discriminant functions of class ω_j and class ω_k , respectively, having positive values. The value of $\rho_{kj}(\cdot)$ ranges between -1 and 1 , and leads to a correct (wrong) decision when it is negative (positive.) In other words, if $\rho_{kj}(\cdot)$ is positive, we can know that \mathbf{x}_n which belongs to class ω_k falls into the wrong region of \mathcal{R}_j .

The function $\ell(\cdot)$ in Eq. (4) is a loss function with respect to the misclassification measure. Any kind of monotonically increasing function can be employed, but in this paper, the following semi-sigmoid function is used ($\xi > 0$):

$$\ell(\rho) = \begin{cases} \frac{1}{1 + \exp(-\xi\rho)} & \text{for } \rho < 0, \\ (\xi\rho + 2)/4 & \text{for } \rho \geq 0. \end{cases} \quad (6)$$

If the ordinary sigmoid function is used, it approaches zero-one function when $\xi \rightarrow \infty$, and then Eq. (4) becomes identical to Eq. (3). However, preliminary experiments show that the above semi-sigmoid loss achieves better performance than the ordinary sigmoid loss on the whole. Therefore, the above semi-sigmoid function is employed in this paper, but further discussion may be needed.

The proposed learning formulation as shown in Eq. (4) can be regarded as a general formulation of classifier design, because it can handle any kind of loss as well as prior probabilities. If we assign unit loss to the wrong decision and no loss to the correct decision, L_{kj} can be written as follows:

$$L_{kj} = 1 - \delta_{kj}, \quad (7)$$

where δ_{kj} is the Kronecker delta. If this zero-one loss is employed, and if we assume that the prior probability is proportional to the number of input vectors, i.e., $P(\omega_k) = N_k/N$, Eq. (4) becomes simpler as follows:

$$L(\boldsymbol{\alpha}) = \frac{1}{N} \sum_{n=1}^N \sum_{k=1}^K \sum_{j \neq k}^K \ell(\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})) 1(\mathbf{x}_n \in \omega_k). \quad (8)$$

This simpler formulation was used in the experiments for UCI machine learning database as shown later.

III. LEARNING OF KERNEL CLASSIFIERS

In this section, a learning method for kernel classifiers is derived on the basis of General Loss Minimization (GLM) described in the previous section. Let us consider the following simple discriminant function for class ω_k :

$$g_k(\mathbf{x}; \boldsymbol{\alpha}) = \mathbf{x}^\top \mathbf{y}_k \quad (9)$$

where \mathbf{y}_k is a reference vector for class ω_k . Here we introduce a nonlinear mapping function $\phi(\mathbf{x})$ that maps

\mathbf{x} from d -dimensional input space to higher order feature space. Then, Eq. (9) can be written as follows:

$$g_k(\mathbf{x}; \boldsymbol{\alpha}) = \phi(\mathbf{x})^\top \phi(\mathbf{y}_k). \quad (10)$$

$\phi(\mathbf{y}_k)$ may be described by a linear combination of mapped input vectors as follows:

$$\phi(\mathbf{y}_k) = \sum_{i=1}^N \alpha_i \phi(\mathbf{x}_i) 1(\mathbf{x}_i \in \omega_k), \quad (11)$$

where $\alpha_i \geq 0$ ($i = 1, \dots, N$) is the classifier parameter to be trained. By substituting the above equation to Eq. (10), we can rewrite it as follows:

$$g_k(\mathbf{x}; \boldsymbol{\alpha}) = \sum_{i=1}^N \alpha_i K(\mathbf{x}, \mathbf{x}_i) 1(\mathbf{x}_i \in \omega_k), \quad (12)$$

where $K(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x})^\top \phi(\mathbf{x}_i)$. $K(\cdot, \cdot)$ is called the Mercer kernel function, and we can compute the inner product of the mapping functions implicitly by calculating the value of the kernel function. In this paper, the following RBF kernel is used:

$$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\gamma \|\mathbf{x} - \mathbf{x}_i\|^2). \quad (13)$$

where $\gamma > 0$ denotes the kernel size.

Finally, by substituting Eq. (12) to Eq. (5), we can formulate the learning of the RBF kernel classifier as follows:

$$\text{minimize } L(\boldsymbol{\alpha}) \quad (14)$$

$$\text{subject to } \sum_{i=1}^N \alpha_i = 1, \quad \forall i, \alpha_i \geq 0 \quad (15)$$

where $L(\boldsymbol{\alpha})$ is defined by Eq. (4). The reason for introducing the constraint is described below. Even though the value of each discriminant function is multiplied by some value, $L(\boldsymbol{\alpha})$ still remains the same value (i.e., $L(k\boldsymbol{\alpha}) = L(\boldsymbol{\alpha})$), because Eq. (5) is defined by the ratio of the discriminant functions. Therefore, $\boldsymbol{\alpha}$ cannot be determined uniquely based on minimizing $L(\boldsymbol{\alpha})$. To solve this problem, the constraint is introduced. In addition, the constraint may contribute to sparseness of $\boldsymbol{\alpha}$, because when the value of one element in $\boldsymbol{\alpha}$ increases, the others decrease.

The minimization of $L(\boldsymbol{\alpha})$ can be conducted based on gradient search. The differential of $L(\boldsymbol{\alpha})$ is derived as shown below.

$$\begin{aligned} \frac{\partial L(\boldsymbol{\alpha})}{\partial \alpha_i} &= \sum_{n=1}^N \sum_{k=1}^K \sum_{j=1}^K \frac{L_{kj} P(\omega_k)}{N_k} \ell'(\rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})) \\ &\quad \times \frac{\partial \rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial \alpha_i} 1(\mathbf{x}_n \in \omega_k), \end{aligned} \quad (16)$$

where $\ell'(\cdot)$ denotes the differential of $\ell(\cdot)$ as follows:

$$\ell'(\rho) = \begin{cases} \xi \ell(\rho)(1 - \ell(\rho)) & \text{for } \rho < 0, \\ \xi/4 & \text{for } \rho \geq 0. \end{cases} \quad (17)$$

The differentiation with respect to α_i depends on the class to which the corresponding \mathbf{x}_i belongs. For α_i which \mathbf{x}_i belongs to class ω_k , it becomes

$$\begin{aligned} \frac{\partial \rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial \alpha_i} &= \frac{\partial \rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial g_k(\mathbf{x}_n; \boldsymbol{\alpha})} \times \frac{\partial g_k(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial \alpha_i} \\ &= \frac{-2g_j(\mathbf{x}_n; \boldsymbol{\alpha})}{(g_j(\mathbf{x}_n; \boldsymbol{\alpha}) + g_k(\mathbf{x}_n; \boldsymbol{\alpha}))^2} \times \alpha_i K(\mathbf{x}_n, \mathbf{x}_i), \end{aligned} \quad (18)$$

and for α_i which \mathbf{x}_i belongs to class ω_j ,

$$\begin{aligned} \frac{\partial \rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial \alpha_i} &= \frac{\partial \rho_{kj}(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial g_j(\mathbf{x}_n; \boldsymbol{\alpha})} \times \frac{\partial g_j(\mathbf{x}_n; \boldsymbol{\alpha})}{\partial \alpha_i} \\ &= \frac{2g_k(\mathbf{x}_n; \boldsymbol{\alpha})}{(g_j(\mathbf{x}_n; \boldsymbol{\alpha}) + g_k(\mathbf{x}_n; \boldsymbol{\alpha}))^2} \times \alpha_i K(\mathbf{x}_n, \mathbf{x}_i). \end{aligned} \quad (19)$$

In the experiments, the conjugate gradient method was used for optimizing the parameters. Before learning, every α_i was set to the same value. In each iteration step, α_i was normalized according to the constraint, and any α_i having sufficiently small value was set to zero. After learning, if α_i was not zero, the corresponding \mathbf{x}_i was used as a reference vector for classification.

IV. EXPERIMENTAL RESULTS

To demonstrate the effectiveness of the proposed learning method, two kinds of experiments were conducted: one was for two-dimensional artificial data, and the other was for the UCI machine learning database. In the proposed method, zero-one loss as shown in Eq. (7) was used in the experiments for simplicity.

A. Two-Dimensional Artificial Data

Two-class two-dimensional artificial data were generated from a Gaussian distribution for each class, and they were shifted horizontally to make them non-linearly separable as shown in Fig. 1. Cross marks and plus marks denote training samples for class ω_1 and class ω_2 , respectively. Three kinds of experiments were conducted with different numbers of training samples: I) 500 samples for each class, II) 50 samples for each class, and III) 50 samples for one class and 500 samples for the other class. In all cases, 500 unseen samples for each class were used for testing.

SVM has two hyperparameters: γ in RBF kernel and C value which is a trade-off between training error and margin. The proposed method also has two hyperparameters: γ in RBF kernel and ξ which controls the slant of the loss function as shown in Eq. (6). The value of these hyperparameters was determined by preliminary experiments.

In Fig. 1, the first and the second columns show the results by SVM and by the proposed method (GLM), respectively. The blue region denotes the decision region for class ω_1 after learning. Likewise, the red region denotes the decision region for class ω_2 . Support vectors by SVM and reference vectors obtained by the proposed method are denoted by white circles in the figure.

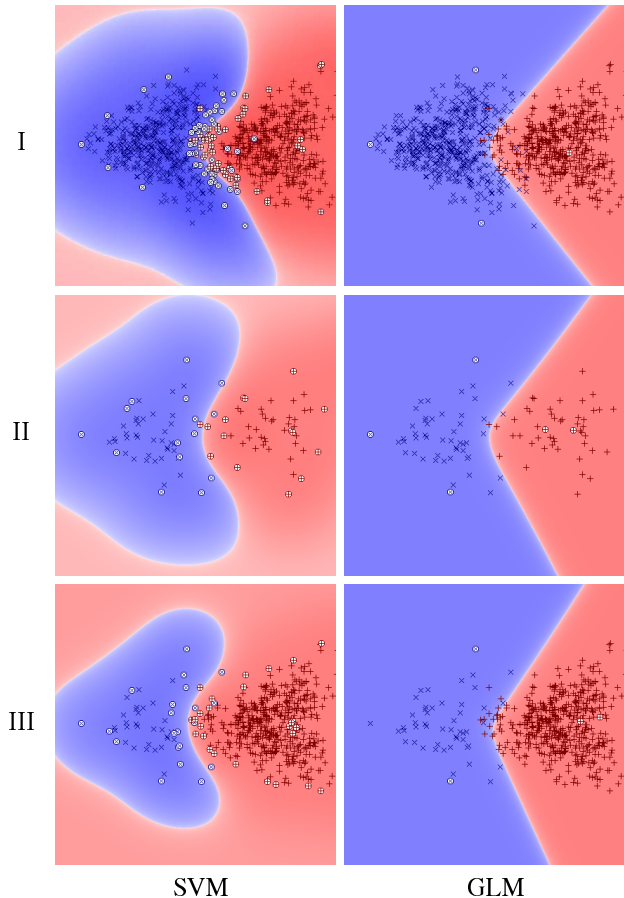


Figure 1. Experimental results for two-dimensional artificial data

Error rates for test data are summarized in Table I. Ideal error rate can be calculated as follows in this case:

$$\begin{aligned} R_{err} &= P(\omega_1) \frac{1}{\sqrt{2\pi}\sigma_1} \int_{0.5}^{\infty} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} dx \\ &\quad + P(\omega_2) \frac{1}{\sqrt{2\pi}\sigma_2} \int_{-\infty}^{0.5} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} dx. \end{aligned} \quad (20)$$

By substituting $P(\omega_1) = P(\omega_2) = 0.5$, $\sigma_1 = \sigma_2 = 0.1$, $\mu_1 = 0.3$, $\mu_2 = 0.7$, we can obtain $R_{err} \simeq 2.3(\%)$. As shown in the table, the results reveal that the proposed method achieves lower error rates than SVM, while the number of reference vectors ($\#RV$ s) obtained by the proposed method is much less than that of support vectors ($\#SV$ s) by SVM. In case III, we may obtain better results for SVM by tuning offset. In the proposed method, however, we can rectify such imbalance properly, without any tuning, by setting the prior probabilities as $P(\omega_1) = P(\omega_2) = 0.5$ in Eq. (4).

B. UCI Machine Learning Database

Experiments for the UCI machine learning database were conducted to evaluate the performance of the proposed

Table I
EXPERIMENTAL RESULTS FOR TWO-DIMENSIONAL ARTIFICIAL DATA

Exp.	SVM		GLM		Ideal
	Error (%)	#SVs	Error (%)	#RVs	
I	2.7	109	2.4	4	2.3
II	3.2	32	2.8	5	2.3
III	7.1	51	2.9	4	2.3

Table II
EXPERIMENTAL RESULTS FOR UCI MACHINE LEARNING DATABASE

DB Name	SVM		GLM	
	Error (%)	#SVs	Error (%)	#RVs
BreastCancer	2.8	65.8	2.6	5.8
HouseVotes84	3.7	101.4	3.9	15.1
Heart1	14.5	159.1	13.5	17.1
Ionosphere	4.8	147.7	8.3	124.1
Sonar	12.1	124.5	12.0	119.2
P.I. Diabetes	22.3	418.1	22.4	16.6
Liver	28.1	202.6	26.3	66.4

method for higher dimensional data. In the experiments, the prior probabilities were assumed to be proportional to the number of samples like in SVM, i.e., Eq. (8) was used for the proposed method. Some data processing was done beforehand as follows: all records containing any missing values were removed from the data set, a binary coding scheme was used for handling categorical variables, and all metric variables were scaled to zero mean and unit variance.

Seven data sets as shown in Table II were evaluated based on 10-fold cross validation. That is to say, the data were divided into 10 partitions, and 9 partitions were used for training and the rest was used for testing. This process was then repeated 10 times, and the obtained results were averaged to produce a single estimation. The hyperparameters were tuned for training data based on 3-fold cross validation with a grid search over the two-dimensional parameter space: for SVM, C ranges from 2^{-5} to 2^{12} and γ ranges from 2^{-10} to 2^5 , and for the proposed method, ξ ranges from 2^1 to 2^7 and γ ranges from 2^{-10} to 2^5 .

Error rates for test data are summarized in Table II. The classification accuracy by the proposed method (GLM) is almost the same as or better than SVM, while the average number of reference vectors is less than 10% of support vectors in the best case. However, the error rate for Ionosphere by the proposed method is much worse than that by SVM. If an exponential loss function is employed, the error rate decreases down to around 6%, but it causes degradation of accuracy for the other data sets. This means that the definition of the loss function is very important for improving classification accuracy depending on the problem.

V. DISCUSSION

SVM is based on structural risk minimization in which the training error is minimized on condition that the margin

is maximized. In the proposed formulation, however, the margin is not maximized while the training error is also minimized. The loss function shown in Eq. (6) works as a hinge loss used in SVM, and the insensitive region is controlled by the value of ξ . Therefore, we can say that the training error is minimized under the *given* margin defined by ξ in the proposed formulation. Therefore, the value of ξ should be tuned carefully or we should pay attention to the definition of the loss function depending on the problem. One possible extension to solve this problem seems to introduce a sort of margin maximization in this formulation.

VI. CONCLUSIONS

A new learning formulation for classifier design called “General Loss Minimization” was proposed based on Bayes decision theory. This formulation is a more general framework of learning which can handle various losses as well as prior probabilities. A learning method for RBF kernel classifiers was derived based on the formulation. Experimental results revealed that the classification accuracy by the proposed method is almost the same as or better than SVM, while the number of reference vectors obtained by the proposed method is less than 10% of support vectors by SVM in the best case. However, the results also revealed that the classification accuracy depends on the definition of the loss function, so we should pay attention to it.

REFERENCES

- [1] A. Sato and K. Yamada, “Generalized Learning Vector Quantization,” *Advances in Neural Information Processing Systems*, 8:423-429, MIT Press, 1996.
- [2] A. Sato, “Discriminative Dimensionality Reduction Based on Generalized LVQ,” *Artificial Neural Networks*, 65-72, LNCS-2130, Springer, 2001.
- [3] C.L. Liu, M. Nakagawa, “Evaluation of Prototype Learning Algorithms for Nearest Neighbor Classifier in Application to Handwritten Character Recognition,” *Pattern Recognition*, 34(3):601-615, 2001.
- [4] C. Cortes and V. Vapnik, “Support Vector Networks,” *Machine Learning*, 20:273-297, 1995.
- [5] A.K. Qin and P.N. Suganthan, “A Novel Kernel Prototype-Based Learning Algorithm,” *Proc. of ICPR’04*, 4:621-624, 2004.
- [6] C. Blake, C. Merz, UCI repository of machine learning databases, University of California, Irvine, Dept. of Information and Computer Sciences 1998, <http://www.ics.uci.edu/~mllearn/>
- [7] B. -H. Juang and S. Katagiri, “Discriminative Learning for Minimum Error Classification,” *IEEE Trans. on Signal Processing*, 40(12):3043-3054, 1992.