Boosting Kernel Discriminant Analysis with Adaptive Kernel Selection

Shinji Kita*, Satoshi Maekawa†, Seiichi Ozawa*, Shigeo Abe*
*Graduate School of Science and Technology, Kobe University, Japan
†National Institute of Information and Communications Technology, Japan
E-mail: {ozawasei,abe}@kobe-u.ac.jp, maekawa@nict.go.jp

Abstract

In this paper, we present a new method to enhance classification performance based on Boosting by introducing nonlinear discriminant analysis as feature selection. To reduce the dependency between hypotheses, each hypothesis is constructed in a different feature space formed by Kernel Discriminant Analysis (KDA). Then, these hypotheses are integrated based on AdaBoost. To conduct KDA in each Boosting iteration within realistic time, a new method of kernel selection is also proposed. Several experiments are carried out for the blood cell data and thyroid data to evaluate the proposed method. The result shows that it is almost the same as the best performance of Support Vector Machine without any time-consuming parameter search.

1 Introduction

Recently, kernel methods have been widely noticed as a powerful approach to solving difficult classification tasks. The Support Vector Machine (SVM) is a typical classifier based on the kernel method. The advantage of kernel methods originally comes from the nonlinear mapping to a high-dimensional feature space. If a proper kernel function is selected, the inputs in the same class can be completely isolated from the others and the class separability is maximized in the feature space. However, the features mapped to such a high-dimensional space often suffer from noise and outliers; hence dimensional reduction for the feature space such as Kernel Principal Component Analysis and Kernel Discriminant Analysis (KDA) have been often used [1][2].

KDA is a promising method of feature selection in which the class separability is maximized in a feature space. However, it is not easy to find a proper kernel function for a particular dataset. To find an optimal kernel, cross-validation has been often employed. However, an eigenvalue problem must be solved at every validation step in KDA; hence, the computation costs often become serious especially when a large training dataset is given. On the other hand, Boosting has been also widely known as a powerful method to realize a strong hypothesis by combining several weak hypotheses [3][4].

From the idea underlying in Boosting, we come upon a new idea for a practical KDA implementation without immense computations; that is, even if week hypotheses are constructed with low-performance features obtained by KDA using a small subset of training samples, we expect to construct a strong hypothesis by combining such week hypotheses based on the Boosting principle. In this framework, we apply KDA to a small number of training samples; hence, it is expected that the computation costs of kernel selection are greatly reduced. Unfortunately, however, if we adopt cross-validation as a kernel selection method, the computation costs are still high because we need to evaluate classification performance at every step in kernel selection and boosting. To overcome this problem, we can adopt another criterion in kernel selection: (between-class scatter)/(within-class scatter).

In this paper, we propose a novel boosting approach in which each weak learner is constructed based on a different feature space whose axes are obtained by KDA with a small subset of training samples. This approach provides a practical implementation for the combination of Boosting and KDA, which can lead to reducing classification dependency between constituent hypotheses

In the next section, we describe our strategy and the Boosting KDA algorithm. Then Section 3 shows some experimental results for two standard datasets.

2 Boosting Kernel Discriminant Analysis

Here, we briefly explain KDA and AdaBoost.M2 [3], and then we propose a novel method to select an appropriate parameter in KDA and a new Boosting strategy to increase the diversity of generated hypotheses. Finally we show the whole learning algorithm of the proposed method.

2.1 KDA and AdaBoost.M2

KDA [2], which is a nonlinear extension of LDA, is well known to give a subspace where the class separability is maximized in a high-dimensional feature space. In KDA, input data are projected into a high-dimensional

feature space at first, then they are projected on a low-dimensional subspace called KDA subspace.

Suppose that a training set is given as $\{(\boldsymbol{x}_{ij}, y_{ij})_{j=1}^{C_i}\}_{i=1}^{C}$ where $\boldsymbol{x}_{ij}$ is an I dimensional column vector and y_{ij} is the class label of $\boldsymbol{x}_{ij}$. C is the number of classes and each class has C_i $(i = 1, \ldots, C)$ samples. Let $N = \sum_i C_i$ be the total number of training samples. We assume that observations have zero-mean in the feature space and this is achievable by adjusting 'Kernel Matrix' (see [5] for details). The inputs are mapped into a high dimensional feature space through a nonlinear mapping function $\phi : \boldsymbol{R}^I \rightarrow F$, where F is the feature space. A between-class scatter matrix B and a within-class scatter matrix W in F are given as follows:

$$B = \frac{1}{N} \sum_{i=1}^{C} C_i \boldsymbol{m}_i \boldsymbol{m}_i^t \tag{1}$$

$$W = \frac{1}{N} \sum_{i=1}^{C} \sum_{j=1}^{C_i} \big(\phi(\boldsymbol{x}_{ij}) - \boldsymbol{m}_i\big)\big(\phi(\boldsymbol{x}_{ij}) - \boldsymbol{m}_i\big)^t \tag{2}$$

where $\boldsymbol{m}_i = \sum_{j=1}^{C_i} \phi(\boldsymbol{x}_{ij})$ is the center of the class i in the feature space. Basis vectors v spanning a KDA subspace are obtained by solving the following eigenvalue problem: $\lambda W v = B v$.

To calculate B and W in the high dimensional space, we use a kernel function $K(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^t \phi(\boldsymbol{x}')$. This enable us to calculate B and W without treating $\phi(\boldsymbol{x})$. This method is well known as 'Kernel Trick' [1]. As one of various kernel functions, the Gaussian kernel

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{\sigma^2}\right) \tag{3}$$

is often used, and we also use this Gaussian kernel here.

On the other hand, AdaBoost is a major boosting algorithm developed by Freund [3], which boosts the performance by the ensemble of weak learners whose performances are slightly better than random guessing. AdaBoost.M2 is one of the highly sophisticated multiclass extensions of AdaBoost, which has been proposed for two-class problems (see [3] for details). We use AdaBoost.M2 to integrate hypotheses.

2.2 Proposed Kernel Selection

In KDA, instead of finding an optimal parameter σ in Eq. (3) by cross-validation, we present a method to select σ such that the following criteria S is maximized:

$$S = \frac{\mathrm{trace}(B)}{\mathrm{trace}(W)}. \tag{4}$$

This criterion is the same as used in the conventional LDA. Applying this kernel selection to several datasets,

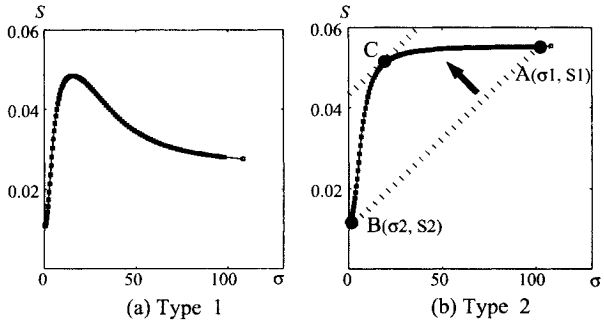

Fig. 1. σ-S curves for Gaussian kernel

we have observed two types of σ-S curves shown in Figs. 1(a),(b). As can be seen from Fig. 1(a), Type 1 has a single peak. In this case, we should select σ at this peak unless it is not too small. If σ is too small, some test samples might be projected to the complementary space of the feature space which is spanned by training data. Consequently, the test performance will get worse even though the training data are correctly classified. To avoid such an inappropriate situation, we should select σ with the largest S under the condition that σ is larger than a threshold value (here we set it to 0.01).

On the other hand, Type 2 in Fig. 1(b) does not have any peak and S monotonically increases to a certain value. Since the parameter σ that maximize S becomes so large, the Gaussian kernel has a similar value in any case; then the kernel matrix can be easily degenerated. Hence, it is preferable to select σ as small as possible under the condition that S is not too small compared with the maximum value. An ad hoc solution for this is to find σ ('C' in Fig. 1(b)) to maximize the following criteria H:

$$H = S - \frac{S_1 - S_2}{\sigma_1 - \sigma_2}\sigma \tag{5}$$

where (σ_1, S_1) and (σ_2, S_2) correspond to the points with minimum and maximum σ, respectively ('A' and 'B' in Fig. 1(b)).

2.3 A Strategy to Generate Diverse Hypotheses

To obtain various KDA subspaces, a small subset of training samples is extracted from the whole training set, then a KDA subspace is constructed from it. Intuitively, this KDA subspace is regarded as a 'weak feature space' because it is constructed based on limited information on training samples. In such a weak feature space, the corresponding hypothesis must also have week performance. Therefore, it is expected that a strong hypothesis is constructed by combining these week hypotheses based on AdaBoost.

For each training subset, r% of training samples are

Input: Training set $\{(\boldsymbol{x}_{ij}, y_{ij})_{j=1}^{C_i}\}_{i=1}^{C}$ where $\boldsymbol{x}_{ij} \in \boldsymbol{R}^I$ and the class label $y_{ij} \in Y = \{1, \ldots, C\}$; the distribution probability of samples $D(i,j) = 1/N$ for $i = 1, \ldots, C$, $j = 1, \ldots, C_i$; the number of iterations T; kernel function K and the percentage r of selected data.

Initialize weight vectors: $w_{ij,y}^1 = D(i,j)/(C-1)$ for $i = 1, \ldots, N$, $j = 1, \ldots, C_i$, $y \in Y - \{y_{ij}\}$

Do for $t = 1, 2, \ldots, T$

1 Set $W_{ij}^t = \sum_{y \neq y_{ij}} w_{ij,y}^t$; $q_t(i,j,y) = \frac{w_{ij,y}^t}{W_{ij}^t}$ for $y \in Y - \{y_{ij}\}$; and set $D_t(i,j) = \frac{W_{ij}^t}{\sum_{i=1}^{C}\sum_{j=1}^{C_i} W_{ij}^t}$

2 Choose r% of the training data randomly, then denote it as R_t.

3 Obtain a kernel parameter σ_t by applying the proposed kernel selection method to R_t.

4 Construct KDA subspace with (K, σ_t, R_t), and project all training data into this subspace, then denote a set of the feature vectors as F_t.

5 With (D_t, q_t, F_t), build the t-th hypothesis h_t : $\boldsymbol{R}^I \times Y \rightarrow [0,1]$

6 Calculate the pseudo-loss of h_t :

$$\epsilon_t = \frac{1}{2}\sum_{i=1}^{C}\sum_{j=1}^{C_i} D_t(i,j)\Big(1 - h_t(\boldsymbol{x}_{ij}, y_{ij}) + \sum_{y \neq y_{ij}} q_t(i,j,y) h_t(\boldsymbol{x}_{ij}, y)\Big)$$

7 Set $\beta_t = \epsilon_t/(1-\epsilon_t)$

8 Set a new weight vector to be $w_{ij,y}^{t+1} = w_{ij,y}^t \beta_t^{(1/2)(1+h_t(\boldsymbol{x}_{ij}, y_{ij}) - h_t(\boldsymbol{x}_{ij}, y))}$ for $i = 1, \ldots, C$, $j = 1, \ldots, C_i$, $y \in Y - \{y_{ij}\}$

Output the final hypothesis

$$h_f(\boldsymbol{x}) = \arg\max_{y \in Y} \sum_{t=1}^{T} \left(\log \frac{1}{\beta_t}\right) h_t(\boldsymbol{x}, y)$$

Fig. 2. The Proposed Boosting KDA Algorithm

randomly chosen. Here, r is set to a small value to reduce the dependency between hypotheses generated in Boosting steps. If r is small, the total combinations of these subsets increase, then it will result in increasing the diversity of hypotheses that leads to high generalization ability [6].

In Fig. 2, we summarize the proposed Boosting KDA algorithm. Step **Initialize**, Steps 1, 5, 6, 7, 8, and Step **Output** correspond to the procedures in AdaBoost.M2. The percentage r of selecting training samples in Step 2 is set to 1, 3, 5, 10, 15 to study the effect of increasing the diversity of hypotheses. The kernel selection in Step 3 is carried out based on the procedure in 2.2 and KDA algorithm stated in 2.1 is conducted in Step 4.

3 Experiments

The proposed Boosting KDA is compared with AdaBoost [7] and SVM in terms of the recognition rate and training speed. In addition, the progress of recognition rates for different r are investigated to study the properties of Boosting KDA. Here, a blood cell dataset [1] and a thyroid dataset [8] are used for the evaluation. The blood cell dataset contains 3097 training data and 3100 test data. The number of classes and attributes are 12 and 13, respectively. The thyroid dataset contains 3772 training data and 3428 test data. The number of classes and attributes are 3 and 21, respectively.

We adopt RBF networks to create hypothesis, and the number of hidden units is set to twice the number of classes. Training of RBF is carried out based on the conjugate gradient descent, which optimizes the centers and widths of radial-basis functions, and modifies the weights to reduce training errors. The maximum iterations in the optimization is set to ten to avoid over-fitting.

Table 1 shows the test performance of the proposed method, AdaBoost, and SVM [1]. As seen from Table 1, the performance of SVM changes depending on the type of kernel and the implementation of multiclass problems. However, the performance of the proposed method is almost the same as the best performance of SVM. Figures 3(a),(b) show the progress of test performance in Boosting KDA as the boosting steps increase (r=1,3,5,10,15). As seen from Figs. 3(a),(b), the performance for larger r

Table 1. The comparison of test performance. For SVM, the best results are picked up from [1].

(a) Blood Cell Data

Algorithm	Kernel	Multiclass Ext.	rate(%)
Boosting KDA	Gauss	AdaBoost.M2	93.74
Boosting Only	—	AdaBoost.M2	91.42
L1/L2 SVM	Gauss	one-against-all	92.77
	Poly	one-against-all	93.58
DDAG SVM	Gauss	pairwise	92.41
	Poly	pairwise	93.00
ECOC SVM	Gauss	(63,10,27)	94.05
	Poly	one-against-all	92.84

(b) Thyroid Data

Algorithm	Kernel	Multiclass Ext.	rate(%)
Boosting KDA	Gauss	AdaBoost.M2	97.90
Boosting Only	—	AdaBoost.M2	96.03
L1/L2 SVM	Gauss	pairwise	97.29
	Poly	pairwise	97.72
DDAG SVM	Gauss	pairwise	97.40
	Poly	pairwise	97.86

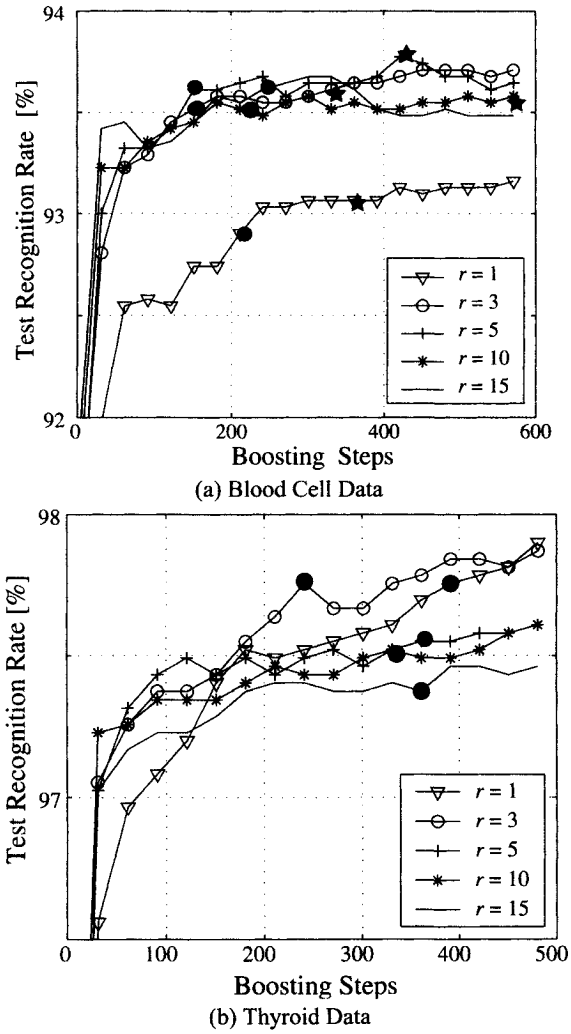

Fig. 3. Test performance of Boosting KDA with r=1,3,5,10,15 for (a) blood cell data and (b) thyroid data. • and ⋆ mean the points where the training performance attains to 99% and 100%, respectively.

is higher than that for smaller r at early boosting steps, while this tendency reverses at late boosting steps. It should be noted that the performance for blood cell data is distinctively degraded when $r = 1$ even if there are no large differences in other cases. Considering that large r leads to slow convergence in training (see the result in Table 2), r should be selected as small as possible within an acceptable performance level.

When it comes to over-fitting, the test performance does not degrade even when the training performance attains to 100% except for the case of $r = 15$ in blood cell data.

Table 2. Average training time (sec.) per iteration in the proposed method (CPU: Intel Pentium-IV, 1.8GHz).

r [%]	1	3	5	10	15
blood cell	10.2	11.6	15.2	54.0	151.8
thyroid	3.5	6.5	16.7	88.9	232.6

4 Conclusions

We have developed a novel method to introduce Kernel Discriminant Analysis into Boosting and a method to choose an appropriate kernel parameter in KDA. The proposed method achieved fairly good performance without any time-consuming parameter tuning. But it takes a little longer time in training than SVM to achieve good recognition performance. To overcome this, a more effective method to choose subsets leading to fast convergence should be developed.

Acknowledgment

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research (B) and (C), and by the Okawa Foundation for Information and Telecommunications.

References

[1] Abe, S. (2005) Support vector machines for pattern classification. Springer

[2] Baudat, G., Anouar, F. (2000) Generalized discriminant analysis using a kernel approach. Neural Computation, Vol. 12: 2385-2404

[3] Freund, Y., Schapire, R. E. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, Vol. 55, No. 1: 119-139

[4] Juwei, L., Plataniotis, K. N., Venetsanopoulos, A. N. (2003) Boosting linear discriminant analysis for face recognition. IEEE Int. Conf. on Image Processing: 14-17

[5] Schölkopf, B., Smola, A., Müller, K. -R. (1996) Nonlinear component analysis as a kernel eigenvalue problem. MPI Technical Report, No. 44

[6] Murua, A. (2002) Upper bounds for error rates of linear combinations of classifiers. IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. 24, No. 5: 591-602

[7] Rätsch, G., Onoda, T., Müller, K. -R. (2001) Soft margins for AdaBoost. Machine Learning, Vol. 42, No. 3: 287-320

[8] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/