Tuning membership functions of kernel fuzzy classifiers by maximizing margins

Morikawa, Kazuya
Ozawa, Seiichi
Abe, Shigeo

(Citation)
Memetic Computing,1(3):221-228

(Issue Date)
2009-11

(Resource Type)
journal article

(Version)
Accepted Manuscript

(URL)
https://hdl.handle.net/20.500.14094/90001052

Noname manuscript No.
(will be inserted by the editor)

Tuning membership functions of kernel fuzzy classifiers by maximizing margins

Kazuya Morikawa · Seiichi Ozawa · Shigeo Abe

Received: date / Accepted: date

Abstract We propose two methods for tuning membership functions of a kernel fuzzy classifier based on the idea of SVM (support vector machine) training. We assume that in a kernel fuzzy classifier a fuzzy rule is defined for each class in the feature space. In the first method, we tune the slopes of the membership functions at the same time so that the margin between classes is maximized under the constraints that the degree of membership to which a data sample belongs is the maximum among all the classes. This method is similar to a linear all-at-once SVM. We call this AAO tuning. In the second method, we tune the membership function of a class one at a time. Namely, for a class the slope of the associated membership function is tuned so that the margin between the class and the remaining classes is maximized under the constraints that the degrees of membership for the data belonging to the class are large and those for the remaining data are small. This method is similar to a linear one-against-all SVM. This is called OAA tuning. According to the computer experiment for fuzzy classifiers based on kernel discriminant analysis and those with ellipsoidal regions, usually both methods improve classification performance by tuning membership functions and classification performance by AAO tuning is slightly better than that by OAA tuning.

Keywords Fuzzy Classifiers · Kernel Discriminant Analysis · Mahalanobis Distance · Membership Functions · Support Vector Machines · Tuning

1 Introduction

Extracting fuzzy rules from data is widely accepted as a method for building fuzzy systems that are comparable in generalization abilities to, and are advantageous in analyzing system behaviors over, neural networks. And various types of fuzzy systems have been developed [1–3].

After fuzzy rule extraction, fuzzy rule tuning, i.e., tuning of membership functions, is one of the important steps in realizing fuzzy systems with high generalization ability. Membership functions can be tuned by multi-layer neural networks [4, 5], genetic

Kazuya Morikawa · Seiichi Ozawa · Shigeo Abe
Graduate School of Engineering, Kobe University, Rokkodai, Nada, Kobe, Japan
E-mail: {ozawasei, abe}@kobe-u.ac.jp

algorithms [6], reducing the total number of misclassifications allowing correctly classified data to be misclassified [7,8], and support-vector-machine like training [9]. In [8], tuning of fuzzy rules of a kernel fuzzy classifier with ellipsoidal regions is analytically done. Namely, by calculating the number of misclassifications if the membership functions are tuned allowing the correctly classified data to be misclassified so long as the total recognition rate is improved. In [9], fuzzy classifiers based on kernel discriminant analysis (KDA) are developed for two-class problems, in which one-dimensional membership functions are defined on the vector that maximally separates two classes in the feature space. Membership functions are tuned by support-vector-machine (SVM) like training to improve generalization ability of the classifier. But this tuning method is restricted to pairwise classification for multi-class problems.

In this paper, we propose two methods for tuning slopes of the membership functions that are applicable to kernel fuzzy classifiers with one fuzzy rule for each class. In the first method, we simultaneously maximize margins between classes under the constraints that the degree of membership of the class to which a data sample belongs is the largest. This is similar to training a linear all-at-once SVM (support vector machine) and is called AAO tuning. To reduce computation time for AAO tuning, in the second method we tune each fuzzy rule separately. Namely, we tune the membership function for a class so that the margin between the class and the remaining classes is maximized under the constraints that the degrees of membership of the data belonging to the class are large while those of the data belonging to the remaining classes are small. This is similar to training a linear one-against-all SVM and is called OAA tuning. By computer experiment, we compare classification performance of the two methods for kernel fuzzy classifiers based on KDA and those with ellipsoidal regions.

In Section 2, we summarize fuzzy classifiers based on KDA and those with ellipsoidal regions and their tuning methods, and in Section 3, we propose two tuning methods. In Section 4, we compare the tuning methods for some benchmark data sets.

2 Kernel fuzzy classifiers

2.1 Architecture

To classify m-dimensional vector $\mathbf{x}$ into one of n classes, we first map the input space into the feature space by the mapping function $\boldsymbol{\phi}(\mathbf{x})$ and generate a fuzzy classifier in the feature space. Assume we have M training data pairs $\{\mathbf{x}_i, y_i\}$ $(i = 1, \ldots, M)$, where $y_i \in \{1, \ldots, n\}$ are class labels. For class i, in the feature space we define a line $\psi_i(\mathbf{x})$, on which a membership function is defined.

We calculate the center of class i data on $\psi_i(\mathbf{x})$, $\mathbf{c}_i$, by

$$\mathbf{c}_i = \frac{1}{|X_i|} \sum_{j \in X_i} \psi_i(\mathbf{x}_j) \quad \text{for} \quad i = 1, ..., n, \tag{1}$$

where X_i is the set of training data indices belonging to class i and $|X_i|$ is the number of elements in X_i.

We define the following fuzzy rule for class i:

$$\text{If } \psi_i(\mathbf{x}) \text{ is } \mathbf{c}_i \quad \text{then } \mathbf{x} \text{ belong to class } \ i. \tag{2}$$

Namely, if $\psi_i(\mathbf{x})$ is closer to $\mathbf{c}_i$, it is more probable that $\mathbf{x}$ belongs to class i.

To calculate the distance from $\psi_i(\mathbf{x})$ to $\mathbf{c}_i$, we define the distance $d_i(\mathbf{x})$, where $d_i(\mathbf{x}) = |\psi_i(\mathbf{x}) - \mathbf{c}_i|$ and based on the distance we define the membership function for class i as follows:

$$m_i(\mathbf{x}) = 1 - \beta_i \, d_i(\mathbf{x}) - b_i, \tag{3}$$

where β_i is a positive parameter to control the slope of the membership function and b_i is a bias term.

Depending on the value of $\psi_i(\mathbf{x})$, (3) may give a negative degree of membership. To avoid this we can use the following equivalent membership function:

$$m_i(\mathbf{x}) = \exp\left(-\beta_i \, d_i^2(\mathbf{x}) - b_i\right). \tag{4}$$

Using the fuzzy rules, $\mathbf{x}$ is classified into the class

$$\arg \max_{i=1,\dots,n} m_i(\mathbf{x}), \tag{5}$$

where arg returns the subscript that maximizes $m_i(\mathbf{x})$.

In a kernel fuzzy classifier based on KDA defined for two-class problems [9], mapped training data are projected onto vector $\mathbf{w}$ in the feature space, where $\mathbf{w}$ is obtained so that the projected data are maximally separated. In this case, for the two classes, $\psi_i(\mathbf{x})$ $(i = 1, 2)$ is given by

$$\psi_i(\mathbf{x}) = \mathbf{w}^T \, \boldsymbol{\phi}(\mathbf{x}). \tag{6}$$

For the kernel fuzzy classifier with ellipsoidal regions [8], $\psi_i(\mathbf{x})$ is given by the kernel Mahalanobis distance $\delta(\mathbf{x})$:

$$\delta^2(\mathbf{x}) = (\boldsymbol{\phi}(\mathbf{x}) - \mathbf{c}_i)^T \, Q^+_{\boldsymbol{\phi}_i} \, (\boldsymbol{\phi}(\mathbf{x}) - \mathbf{c}_i), \tag{7}$$

where $\mathbf{c}_i$ is the center of class i data in the feature space and $Q^+_{\boldsymbol{\phi}_i}$ is the pseudo-inverse of the covariance matrix $Q_{\boldsymbol{\phi}_i}$. The membership function for class i is given by

$$m_i(\mathbf{x}) = \exp\left(-\beta_i \, \delta^2(\mathbf{x}) - b_i\right), \tag{8}$$

where β_i is a positive parameter to control the Mahalanobis distance and b_i is a bias term.

In the following, we briefly explain the kernel fuzzy classifier based on KDA and those with ellipsoidal regions and their tuning methods.

2.2 Kernel fuzzy classifiers based on KDA

2.2.1 Determination of $\mathbf{w}$

In the KDA-based fuzzy classifier, $\psi_i(\mathbf{x})$, on which the membership function is defined, is given by (6). Vector $\mathbf{w}$ is determined by KDA so that two classes are maximally separated in the feature space. Namely, $\mathbf{w}$ is obtained by maximizing

$$J(\mathbf{w}) = \frac{d_c^2}{s^2} = \frac{\mathbf{w}^T Q_B \mathbf{w}}{\mathbf{w}^T Q_T \mathbf{w}}, \tag{9}$$

where Q_B is the between-class scatter matrix given by

$$Q_B = (\mathbf{c}_1 - \mathbf{c}_2)(\mathbf{c}_1 - \mathbf{c}_2)^T, \tag{10}$$

and Q_T is the total scatter matrix given by

$$Q_T = \frac{1}{M}(\boldsymbol{\phi}(\mathbf{x}_1), ..., \boldsymbol{\phi}(\mathbf{x}_M))(I_M - \mathbf{1}_M) \begin{pmatrix} \boldsymbol{\phi}^T(\mathbf{x}_1) \\ \vdots \\ \boldsymbol{\phi}^T(\mathbf{x}_M) \end{pmatrix}. \tag{11}$$

Here, I_M is the $M \times M$ unit matrix and $\mathbf{1}_M$ is the $M \times M$ matrix with all elements being $1/M$. Maximizing (9) minimizes the variance of all the mapped training data measured on $\mathbf{w}$ while maximizing the distance between classes measured on $\mathbf{w}$.

We calculate $\mathbf{w}$ using kernel tricks. Any solution $\mathbf{w}$ in the feature space can be written as an expansion of the form

$$\mathbf{w} = (\boldsymbol{\phi}(\mathbf{x}_1), ..., \boldsymbol{\phi}(\mathbf{x}_M))\boldsymbol{\alpha}, \tag{12}$$

where $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_M)^T$ and $\alpha_1, ..., \alpha_M$ are scalars. Substituting (12) into (9), we can rewrite the KDA criterion, J, as

$$J(\boldsymbol{\alpha}) = \frac{\boldsymbol{\alpha}^T K_B \boldsymbol{\alpha}}{\boldsymbol{\alpha}^T K_T \boldsymbol{\alpha}}, \tag{13}$$

where

$$K_B = (\mathbf{k}_{B_1} - \mathbf{k}_{B_2})(\mathbf{k}_{B_1} - \mathbf{k}_{B_2})^T, \tag{14}$$

$$\mathbf{k}_{B_i} = \begin{pmatrix} \frac{1}{|X_i|} \sum_{j \in X_i} H(\mathbf{x}_1, \mathbf{x}_j) \\ \cdots \\ \frac{1}{|X_i|} \sum_{j \in X_i} H(\mathbf{x}_M, \mathbf{x}_j) \end{pmatrix} \quad \text{for} \quad i = 1, 2, \tag{15}$$

$$K_T = \frac{1}{M} K (I_M - \mathbf{1}_M) K. \tag{16}$$

Here $H(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}^T(\mathbf{x})\boldsymbol{\phi}(\mathbf{x}')$ is a kernel function, and $K = \{H(\mathbf{x}_i, \mathbf{x}_j)\}$ is a kernel matrix constructed by using all the training data. Thus K_T is a positive semi-definite matrix. If K_T is positive definite, the solution of (13) is given by

$$\boldsymbol{\alpha} = K_T^{-1}(\mathbf{k}_{B_1} - \mathbf{k}_{B_2}). \tag{17}$$

If K_T is positive semi-definite, the inverse K_T^{-1} does not exist. One way to overcome singularity is to add positive values to the diagonal elements [10]:

$$\boldsymbol{\alpha} = (K_T + \varepsilon I_M)^{-1}(\mathbf{k}_{B_1} - \mathbf{k}_{B_2}), \tag{18}$$

where ε is a small positive parameter.

Assuming that $\|\mathbf{w}\| = 1$, we can calculate the projection of $\boldsymbol{\phi}(\mathbf{x})$ on $\mathbf{w}$, p, with kernel tricks as follows:

$$p = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}) = (H(\mathbf{x}, \mathbf{x}_1), ..., H(\mathbf{x}, \mathbf{x}_M))\boldsymbol{\alpha}. \tag{19}$$

2.2.2 Tuning membership functions

We define the function $L(\mathbf{x})$ as the difference of $m_1(\mathbf{x})$ and $m_2(\mathbf{x})$:

$$\begin{aligned} L(\mathbf{x}) &= m_1(\mathbf{x}) - m_2(\mathbf{x}) \\ &= -\frac{d_1(p)}{\beta_1} + \frac{d_2(p)}{\beta_2} + (b_2 - b_1). \end{aligned} \tag{20}$$

Now we set $\boldsymbol{\beta} = (-\frac{1}{\beta_1}, \frac{1}{\beta_2})^T$, $\mathbf{d} = (d_1(p), d_2(p))^T$, and $b = b_2 - b_1$. Then (20) is rewritten as follows:

$$L(\mathbf{d}) = \boldsymbol{\beta}^T \mathbf{d} + b. \tag{21}$$

If the data sample $\mathbf{x}$ satisfies $L(\mathbf{d}) > 0$, it is classified into Class 1, and if $L(\mathbf{d}) < 0$, it is classified into Class 2. When $L(\mathbf{d}) = 0$, it is unclassifiable.

From the above discussion, $L(\mathbf{d})$ is the decision function of the fuzzy classifier whose input is a two-dimensional vector $\mathbf{d} = (d_1(p), d_2(p))^T$, and its form is the same as that of the decision function of linear SVMs. Hence, by calculating the weight vector $\boldsymbol{\beta}$ and bias term b based on the same training algorithm as that of linear SVMs in the two-dimensional space $(d_1(p), d_2(p))$, we can determine the slope parameter β_i and bias term b_i of each membership function. (But in this formulation, we cannot determine the values of b_1 and b_2 uniquely. However, because the classification boundary is invariant so long as $b = b_2 - b_1$ is constant, we can assume that either b_i is equal to 0.)

For multiclass problems, if we define fuzzy rules by one-against-all strategy, we need to define a fuzzy rule for a class and a fuzzy rule for the remaining classes. Namely, we first determine the vector that maximally separates class i and the remaining classes. Then for class i, we define a fuzzy rule and for classes other than i we define one fuzzy rule. But this is not a good strategy because it is difficult to interpret the fuzzy rules for the ensembles of classes.

But by pairwise classification, we can apply the above method. Namely, we first determine the vector that maximally separates the two classes in a class pair and tune the membership function defined on the vector. In this case, we need the bias terms to be zero, because the above method cannot determine the bias terms uniquely.

2.3 Kernel fuzzy classifiers with ellipsoidal regions

In calculating the kernel Mahalanobis distance in (8), we need to use kernel tricks [8] but it is time consuming. Therefore, we consider calculating the kernel Mahalanobis distance using the idea of the empirical feature space [11,12], which is spanned by at most M training data and is equivalent to the feature space in that they give the same kernel values if one of the argument is included in the training data. To speedup computation, we avoid calculating the eigenvalues of the kernel matrix and use the following mapping function to the empirical feature space [12]:

$$\mathbf{h}(\mathbf{x}) = \left(H(\mathbf{x}_{k_1}, \mathbf{x}), ..., H(\mathbf{x}_{k_N}, \mathbf{x})\right)^T, \tag{22}$$

where $\mathbf{x}_{k_j}$ $(j = 1, ..., N)$ are linearly independent in the feature space. The linearly independent data can be selected by Cholesky factorization.

Using (22), the center $\mathbf{c}_i$ for class i in the empirical feature space is calculated by

$$\mathbf{c}_i = \frac{1}{|X_i|} \sum_{j \in X_i} \mathbf{h}(\mathbf{x}_j). \tag{23}$$

Then the covariance matrix $Q_{\boldsymbol{\phi}_i}$ is given by

$$Q_{\boldsymbol{\phi}_i} = \frac{1}{|X_i|} \sum_{j \in X_i} (\mathbf{h}(\mathbf{x}_j) - \mathbf{c}_i)^T (\mathbf{h}(\mathbf{x}_j) - \mathbf{c}_i). \tag{24}$$

Tuning parameters β_i in (8) are tuned until there is no improvement in the recognition rate. Namely, if β_i is decreased, the degree of $m_i(\phi(\boldsymbol{x}))$ increases. This may lead to correct classification of the data, which were misclassified before tuning, while some data, which were correctly classified, may be misclassified. Here we allow the data, which were classified correctly before tuning to be misclassified as long as the overall recognition rate of the training data is improved. We determine the values of β_i, in which misclassification or correct classification of training data occurs, dived the range of β_i accordingly, calculate the number of a net increase in correct classification, and change the value of β_i to the middle of the interval that realizes the maximum recognition rate. This tuning method is efficient in tuning but because it is a local optimization method, we may obtain a local optimum solution.

3 Proposed tuning methods

In this section, we discuss two methods for tuning membership functions: AAO (all-at-once) tuning and OAA (one-against-all) tuning methods.

3.1 All-at-once tuning methods

For $\mathbf{x}$ belonging to class i if

$$m_i(\mathbf{x}) > m_j(\mathbf{x}) \quad \text{for} \quad j \neq i, j = 1, \ldots, n \tag{25}$$

are satisfied, $\mathbf{x}$ is correctly classified. This is equivalent to

$$\beta_i \nu_i(\mathbf{x}) + b_i < \beta_j \nu_j(\mathbf{x}) + b_j \quad \text{for} \quad j \neq i, j = 1, \ldots, n, \tag{26}$$

where $\nu_i(\mathbf{x}) = d_i(\mathbf{x})$ for the kernel fuzzy classifier based on KDA and $\nu_i(\mathbf{x}) = \delta_i^2(\mathbf{x})$ for the kernel fuzzy classifier with ellipsoidal regions.

We consider tuning membership functions so that the margin between classes are maximized under the constraints (26). Because $\nu_i(\mathbf{x})$ is a constant for training data, (26) are linear inequalities. Thus, we can formulate the optimization of β_i and b_i using all-at-once formulation of SVMs with linear kernels [13]. We call this tuning method AAO tuning.

We consider solving a linear programming support vector machine as follows:

$$\text{minimize} \qquad Q(\boldsymbol{\beta}, \mathbf{b}, \boldsymbol{\xi}) = \sum_{i=1}^{n} \beta_i + C \sum_{j=1}^{M} \sum_{i \neq y_j, i=1}^{n} \xi_{ji} \tag{27}$$

$$\text{subject to} \qquad \beta_i \, \nu_i(\mathbf{x}_j) - \beta_{y_j} \, \nu_{y_j}(\mathbf{x}_j) + b_i - b_{y_j} \geq 1 - \xi_{ji}, \tag{28}$$

$$\beta_i > 0 \text{ for } \quad i \neq y_j, \ i = 1, ..., n, \ j = 1, ..., M, \tag{29}$$

where C is a margin parameter that determines the trade-off between the classification error and the generalization ability, ξ_{ji} are non-negative slack variables for $\mathbf{x}_j$ and class i, and $y_j \in \{1, \ldots, n\}$ is a class label. The first term in the objective function is to maximize the margin. By taking the linear sum of β_i, instead of the quadratic term, the above problem become a linear programming problem. The constant 1 in the right hand side of (28) is to introduce the margin between classes.

To solve (27) and (29) by linear programming, we need to change the variables nonnegative. Thus, defining $b_i^+ \geq 0$, $b_i^- \geq 0$, we can express $b_i = b_i^+ - b_i^-$ and the following linear programming program is obtained:

$$\text{minimize} \quad Q(\boldsymbol{\beta}^+, \boldsymbol{\beta}^-, b^+, b^-, \boldsymbol{\xi}) = \sum_{i=1}^{n} \beta_i + C \sum_{j=1}^{M} \sum_{i \neq y_j, i=1}^{n} \xi_{ji} \tag{30}$$

$$\begin{aligned} \text{subject to} \quad & \beta_i \, \nu_i(\mathbf{x}_j) - \beta_{y_j} \nu_{y_j}(\mathbf{x}_j) + b_i^+ - b_i^- - b_{y_j}^+ + b_{y_j}^- \geq 1 - \xi_{ji}, \\ & \beta_i > 0 \quad \text{for} \quad i \neq y_j, \ i = 1, ..., n, \ j = 1, ..., M. \end{aligned} \tag{31}$$

The number of variables for (30) and (31) is $3n+M(n-1)$ and the number of constraints is $M(n-1)$.

3.2 One-against-all tuning

The optimization problem given by (30) and (31) are inefficient to solve, if the number of training data and/or the number of classes are large. To solve the optimization problem in such a situation, we consider approximating the all-at-once formulation given by (30) and (31) by the one-against-all formulation called OAA tuning. To do this, we impose the following conditions to $\mathbf{x}_j$ $(j = 1, \ldots, M)$:

$$\beta_i \, \nu_i(\mathbf{x}_j) + b_i \leq 0.5 + \xi_{ij} \qquad \text{for} \quad y_{ij} = 1, \tag{32}$$

$$\beta_i \, \nu_i(\mathbf{x}_j) + b_i \geq 1 - \xi_{ij} \qquad \text{for} \quad y_{ij} = -1, \tag{33}$$

where $y_{ij} = 1$ if $\mathbf{x}_j$ belongs to class i and -1 otherwise, and $\xi_{ij} (\geq 0)$ are slack variables associated with $\mathbf{x}_j$ for class i. Inequality (32) is to make the degree of membership to which the data belongs higher and (33) is to make the degree to which the data does not belong lower.

Combining (32) and (33), we obtain

$$y_{ij}(\beta_i \, \nu_i(\mathbf{x}_j) + b_i) \leq p_{ij} + \xi_{ij} \qquad \text{for} \quad j = 1, \ldots, M, \tag{34}$$

where $p_{ij} = 0.5$ for $y_{ij} = 1$ and -1 for $y_{ij} = -1$.

Then, for class i we solve the following problem:

$$\text{minimize} \qquad Q(\beta_i, b_i, \boldsymbol{\xi}_i) = \beta_i + C \sum_{j=1}^{M} \xi_{ij} \tag{35}$$

$$\text{subject to} \qquad y_{ij}(\beta_i \, \nu_i(\mathbf{x}_j) + b_i) \leq p_{ij} + \xi_{ij}, \quad \beta_i > 0 \text{ for} \quad j = 1, ..., M. \tag{36}$$

Like (30) and (31), defining $b_i^+ \geq 0$, $b_i^- \geq 0$ for (35) and (36), we obtain $b_i = b_i^+ - b_i^-$. Thus, (35) and (36) become

$$\text{minimize} \quad Q(\beta_i^+, \beta_i^-, b_i^+, b_i^-, \boldsymbol{\xi}) = \beta_i + C \sum_{j=1}^{M} \xi_{ij} \tag{37}$$

$$\text{subject to } y_{ij}\Big(\beta_i \nu_{ij}(\mathbf{x}_j) + b_i^+ - b_i^-\Big) \leq p_{ij} + \xi_{ij},$$
$$\beta_i > 0 \text{ for } j = 1, ..., M. \tag{38}$$

We solve the above problem for $i = 1, \ldots, n$. Thus, for each class the number of variables is $M + 3$ and the number of constraints is M.

The optimization problem given by (37) and (38) can be solved much faster than that by (30) and (31) but for a large sized problem, we need speeding up tuning using the decomposition technique [14].

4 Computer experiments

We compared the effectiveness of the proposed methods using two-class [15] and multi-class problems [13]. We used the kernel fuzzy classifier based on KDA [9] and kernel fuzzy classifier with ellipsoidal regions [8] and tuned the membership functions using the two proposed methods. The fuzzy classifier based on KDA has a parameter ε to determine the threshold to select independent data and in our study we set $\varepsilon = 10^{-8}$. As a baseline classifier, we used fuzzy pairwise L1 support vector machine [13,16]. We used the simplex method to solve a linear programming SVM.

4.1 Benchmark data sets

Table 1 lists the benchmark datasets used in our study. For two class problems, there are 100 or 20 pairs of training and test data sets. Except for the wine and glass data sets, each multi-class problem consists of one training data set and one test data set. Therefore, for these problems with test data sets, we compared the recognition rates of the test data sets. For the wine and glass data sets, we compared the recognition rates of the validation data sets in fivefold cross-validation, namely, the recognition rates of the data that were deleted during cross-validation.

4.2 Parameter setting

We used RBF kernels: $\exp(-\gamma\|\mathbf{x} - \mathbf{x}'\|^2)$ where γ is a positive kernel parameter, and determined the values of kernel parameter and the margin parameter by fivefold cross-validation. For two-class problems we determined the values using the first five training data sets. The parameter ranges are as follows: $\gamma = \{0.1, 0.5, 1, 5, 10, 15\}$, $C = \{1, 10, 50, 100, 500, 1000, 2000, 3000, 5000, 8000, 10000, 50000, 100000\}$. We used the SVM as a baseline classifier and Table 2 shows the parameter values for the SVM and the kernel fuzzy classifier based on KDA. Table 3 shows the selected parameter values for the kernel fuzzy classifier with ellipsoidal regions.

Table 1 Benchmark data sets

Data	Sets	Inputs	Train	Test	Classes
Banana	100	2	400	4900	2
B. cancer	100	9	200	77	2
Diabetes	100	8	468	300	2
German	100	20	700	300	2
Heart	100	13	170	100	2
Image	20	18	1300	1010	2
Ringnorm	100	20	400	7000	2
F. solar	100	9	666	400	2
Splice	20	60	1000	2175	2
Thyroid	100	5	140	75	2
Titanic	100	3	150	2051	2
Twonorm	100	20	400	7000	2
Waveform	100	21	400	4600	2
Iris	1	4	75	75	3
Numeral	1	12	810	820	10
Segmentation	1	19	210	2100	7
Vehicle	1	18	188	658	4
Wine	1	13	178	—	3
Glass	1	9	214	—	6

Table 2 Parameter values for the SVM and the kernel fuzzy classifier based on KDA

	SVM		OAA-Tuning		AAO-Tuning	
Data	γ	C	γ	C	γ	C
Banana	15	100	10	50	10	10
B. cancer	0.1	500	0.1	3000	0.1	10
Diabetes	0.1	3000	0.5	10	0.1	50
German	0.1	50	0	0	0.1	10
Heart	0.1	50	0.1	50	0.1	100
Image	15	500	0	0	15	1
Ringnorm	15	1	0.1	500	0.1	500
F. solar	0.5	10	15	100	0.1	10
Splice	10	10	10	1	10	1
Thyroid	15	100	5	100	1	1000
Titanic	0.5	10	1	10	15	1
Twonorm	0.5	1	0.5	50	0.5	1
Waveform	10	1	0.5	50	0.5	10
Iris	0.1	100	1	10	1	10
Numeral	0.1	500	0.5	50	0.5	500
Segmentation	10	100	10	50	1	1
Vehicle	0.5	10000	5	10	10	1
Wine	5	500	0.5	1	0.5	10
Glass	15	50	10	50	10	10

4.3 Results for kernel fuzzy classifiers based on KDA

Table 4 lists the results of the proposed tuning methods for the kernel fuzzy classifier based on KDA. We also include the results for the SVM and fuzzy classifier based on KDA without tuning. Namely, we set $\alpha_i = 1$ and $b_i = 0$. If the kernel parameter value was different for OAA tuning and AAO tuning, we used the parameter value with the better recognition rate between the tuning methods. For the two-class problems we show the recognition rates and the standard deviations. But for the multiclass

Table 3 Parameter values for the kernel fuzzy classifier with ellipsoidal regions

	OAA-Tuning		AAO-Tuning	
Data	γ	C	γ	C
Banana	10	1	15	1
B. cancer	0.1	1	0.1	1
Diabetes	15	1	0.5	1
German	15	1	0.1	1
Heart	0.1	1	0.1	1
Image	10	1	10	1
Ringnorm	0.1	1	0.1	1
F. solar	0.5	1	0.5	1
Splice	15	1	0.1	1
Thyroid	1	1	1	1
Titanic	5	1	1	1
Twonorm	15	1	0.1	1
Waveform	10	1	0.5	1
Iris	0.1	10	0.1	50
Numeral	0.1	1	—	—
Segmentation	0.1	10	0.1	1
Vehicle	0.5	1	0.1	10
Wine	5	1	0.1	1
Glass	1	1	0.5	10

problems, the recognition rates for the test data are shown except for the wine and glass data sets. For these data sets, the recognition rates for the cross-validation data sets are shown.

For the two-class problems, we performed Welch t test with the significance level of 5% and showed the best results by the statistical test in boldface. Likewise, for the multiclass problems, the best recognition rate is shown in boldface.

From the table, the performance of the AAO tuning method is comparable to or better than that of the SVM six times. And there is not much performance difference between OAA tuning and AAO tuning. By tuning the fuzzy classifier both or either by OAA tuning or AAO tuning, the recognition rate is improved or comparable compared to that without tuning.

For multi-class problems, the recognition rates by AAO tuning are slightly lower than those by SVM. To check the effect of ε on the recognition rate, we changed the value of ε, fixing the values of γ and C determined for $\varepsilon = 10^{-8}$, and evaluated the recognition rate of the iris test data set. Table 5 shows the results. For $\varepsilon = 10^{-4}$ and 10^{-6} the best recognition rate of 97.33% was obtained. Thus, there is still a room for improving the recognition rate by determining the optimal value of ε by cross-validation.

As discussed in Section 3, AAO tuning requires more computation time and memory than OAA tuning. We compared the computation time of the tuning methods including training and testing the classifier for the parameter values determined by cross-validation. Table 6 shows the computational time for the iris and numeral data sets by AAO tuning and OAA tuning. From the table, the optimization problem by OAA tuning can be solved much faster than that by AAO tuning especially for the numeral data set with 10 classes.

4.4 Results for fuzzy classifiers with ellipsoidal regions

Table 7 lists the recognition rates of the kernel fuzzy classifier with ellipsoidal regions. The same as Table 4, the best recognition rates are shown in boldface. For the titanic problem, the average recognition rate by the AAO tuning is shown in boldface and the standard deviation by the SVM is shown in boldface. This means that the average recognition rate by the AAO tuning is statistically better than that by the SVM but the standard deviation by the AAO tuning is statistically larger than that by the SVM.

The two tuning methods performed comparable to or better than the SVM two or three times. In general, the recognition performance is inferior to the kernel fuzzy classifier based on KDA. But by tuning the fuzzy rules both or either by AAO tuning or OAA tuning, the recognition rate is improved. Thus, the effectiveness of the tuning methods is also proved for this classifier.

Table 4 Recognition rates of the test data by the kernel fuzzy classifier based on KDA

Data	SVM	OAA Tuning	AAO Tuning	Without Tuning
Banana	**89.3±0.52**	89.1±0.55	88.0±0.65	89.1±0.53
B. cancer	**72.4±4.67**	**73.8±4.55**	**73.5±4.81**	69.4±4.43
Diabetes	**76.3±1.83**	75.3±3.47	**75.8±1.85**	75.2±1.96
German	**76.2±2.27**	75.3±2.16	75.4±2.41	72.4±2.48
Heart	**83.7±3.41**	82.4±3.19	82.3±3.28	82.3±3.20
Image	**97.3±0.41**	**97.2±0.40**	**97.2±0.38**	**97.2±0.34**
Ringnorm	97.8±0.30	**97.9±0.27**	97.7±0.35	94.7±0.60
F. solar	**67.6±1.74**	60.1±5.29	66.3±1.55	66.3±1.58
Splice	**89.2±0.71**	**89.3±0.61**	**89.3±0.61**	**89.3±0.61**
Thyroid	**96.1±2.08**	95.0±2.30	92.9±3.11	94.6±2.35
Titanic	**77.2±1.12**	72.8±3.36	77.1±1.86	76.6±2.04
Twonorm	**97.6±0.14**	97.3±0.21	97.3±0.22	97.3±0.21
Waveform	90.0±0.44	**90.3±0.44**	**90.3±0.41**	90.1±0.49
Iris	**97.33**	94.67	94.67	89.33
Numeral	**99.63**	99.39	99.51	98.41
Segmentation	**93.81**	90.24	90.76	88.62
Vehicle	**82.98**	74.47	77.36	75.38
Wine	**100**	99.43	**100**	99.43
Glass	**71.48**	71.04	70.58	66.32

Table 5 Recognition rates of the iris test data set for different values of ε

ε	Test
10^{-1}	90.67
10^{-2}	90.67
10^{-3}	94.67
10^{-4}	**97.33**
10^{-5}	94.67
10^{-6}	**97.33**
10^{-7}	96.00
10^{-8}	94.67

Table 6 Computation time (seconds)

Data	OAA Tuning	AAO Tuning
Iris	0.121	0.188
Numeral	220	2708

5 Conclusions

In this paper we discuss two methods of tuning membership functions. The first method tunes membership functions all at once by an SVM with linear kernels. We call this method AAO tuning. And the second method tunes the membership function for a class by a one-against-all SVM. We call this method OAA tuning. According to the computer experiment on classification performance of kernel fuzz classifiers based on KDA and those with ellipsoidal regions, both tuning methods improved classification performance by tuning and that of AAO tuning is slightly better than that of OAA tuning. Comparing the fuzzy classifier based on KDA and the fuzzy classifier with ellipsoidal regions, usually the former performed better.

Comparing the results of the proposed methods with those of the SVM, the SVM performed better in most cases, especially for the fuzzy classifier with ellipsoidal regions. We would like to leave the problem of clarifying the reason as a future study.

References

1. L. I. Kuncheva. *Fuzzy Classifier Design*. Physica-Verlag, Heidelberg, Germany, 2000.
2. R. Fullér. *Introduction to Neuro-Fuzzy Systems*. Physica-Verlag, New York, N. Y., 2000.
3. S. Abe. *Pattern Classification: Neuro-Fuzzy Methods and Their Comparison*. Springer-Verlag, London, 2001.
4. C.-T. Lin and C.-F. Juang. An adaptive neural fuzzy filter and its applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 27(4):635–656, 1997.
5. J. Kim and N. Kasabov. HyFIS: Adaptive neuro-fuzzy inference systems and their application to nonlinear dynamical systems. *Neural Networks*, 12(9):1301–1319, 1999.
6. H. Ishibuchi, T. Nakashima, and T. Murata. Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 29(5):601–618, 1999.
7. S. Abe and R. Thawonmas. A fuzzy classifier with ellipsoidal regions. *IEEE Transactions on Fuzzy Systems*, 5(3):358–368, 1997.

Table 7 Recognition rates by the kernel fuzzy classifier with ellipsoidal regions

Data	SVM	OAA-Tuning	AAO-Tuning	Without Tuning
Banana	**89.3±0.52**	**89.4±0.47**	89.1±0.54	88.6±0.52
B. cancer	**72.4±4.67**	**73.5±4.71**	**73.6±4.32**	68.8±4.62
Diabetes	**76.3±1.83**	71.3±2.48	75.1±1.80	72.2±2.56
German	**76.2±2.27**	73.6±3.41	75.2±2.24	71.3±2.26
Heart	**83.7±3.41**	81.6±3.55	81.4±3.31	80.5±3.47
Image	**97.3±0.41**	97.0±0.83	96.8±0.90	89.6±1.51
Ringnorm	97.8±0.30	97.4±1.24	**97.9±0.35**	50.4±1.13
F. solar	**67.6±1.74**	60.8±5.15	64.7±2.29	65.5±1.62
Splice	**89.2±0.71**	86.6±1.69	86.1±1.07	74.0±2.40
Thyroid	**96.1±2.08**	95.3±2.19	95.0±2.16	73.7±7.04
Titanic	77.2±**1.12**	74.9±5.36	**77.8**±1.38	75.7±2.09
Twonorm	**97.6±0.14**	95.2±4.33	97.2±0.33	96.8±0.27
Waveform	**90.0±0.44**	86.6±2.08	87.1±1.06	78.9±1.80
Iris	**97.33**	94.67	93.33	**97.33**
Numeral	**99.63**	99.15	—	99.51
Segmentation	**93.81**	86.86	73.38	90
Vehicle	**82.98**	80.70	81.16	76.29
Wine	**100**	99.44	96.62	99.43
Glass	**71.48**	68.69	69.15	63.12

8. K. Kaieda and S. Abe. KPCA-based training of a kernel fuzzy classifier with ellipsoidal regions. *International Journal of Approximate Reasoning*, 37(3):145–253, 2004.
9. R. Hosokawa and S. Abe. Fuzzy classifiers based on kernel discriminant analysis. In J. Marques de Sá, L. A. Alexandre, W. Duch, and D. Mandic, editors, *Artificial Neural Networks (ICANN 2007)—Proceedings of the Seventeenth International Conference, Porto, Portugal, Part II*, pages 180–189. Springer-Verlag, Berlin, Germany, 2007.
10. S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. Fisher discriminant analysis with kernels. In Y.-H. Hu, J. Larsen, E. Wilson, and S. Douglas, editors, *Neural Networks for Signal Processing IX—Proceedings of the 1999 IEEE Signal Processing Society Workshop*, pages 41–48, 1999.
11. H. Xiong, M. N. S. Swamy, and M. O. Ahmad. Optimizing the kernel in the empirical feature space. *IEEE Transactions on Neural Networks*, 16(2):460–474, 2005.
12. S. Abe. Sparse least squares support vector training in the reduced empirical feature space. *Pattern Analysis and Applications*, 10(3):203–214, 2007.
13. S. Abe. *Support Vector Machines for Pattern Classification*. Springer-Verlag, London, 2005.
14. Y. Torii and S. Abe. Decomposition techniques for training linear programming support vector machines. *Neurocomputing*, 72(4-6):973–984, 2009.
15. Intelligent Data Analysis Group. http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm.
16. A. Asuncion and D. J. Newman. UCI machine learning repository, http://www.ics.uci.edu/∼mlearn/MLRepository.html. 2007.