Fuzzy Least Squares Support Vector Machines for Multiclass Problems

Daisuke Tsujinishi
Graduate School of Science and Technology
Kobe University
Rokkodai, Nada, Kobe, Japan
Email: tujinisi@chevrolet.eedept.kobe-u.ac.jp

Shigeo Abe
Graduate School of Science and Technology
Kobe University
Rokkodai, Nada, Kobe, Japan
Email: abe@eedept.kobe-u.ac.jp

***Abstract*— In least squares support vector machines (LS-SVMs), the optimal separating hyperplane is obtained by solving a set of linear equations instead of solving a quadratic programming problem. But since SVMs and LS-SVMs are formulated for two-class problems, unclassifiable regions exist when they are extended to multiclass problems.**

In this paper, we discuss fuzzy least squares support vector machines that resolve unclassifiable regions for multiclass problems. We define a membership function in the direction perpendicular to the optimal separating hyperplane that separates a pair of classes. Using the minimum or average operation for these membership functions, we define a membership function for each class. Using some benchmark data sets, we show that recognition performance of fuzzy LS-SVMs with the minimum operator is comparable to that of fuzzy SVMs, but fuzzy LS-SVMs with the average operator showed inferior performance.

I. Introduction

Support vector machines proposed by Vapnik (1998) are trained by solving a quadratic optimization problem. Least squares SVMs proposed by Suykens and Vandewalle (1999) are trained by solving a set of linear equations. But since SVMs and LS-SVMs are formulated for two-class classification problems, an extension to multiclass problems is not unique. There are roughly four types of support vector machines that handle multiclass problems:

1) one-against-all support vector machines (Vapnik, 1998),
2) pairwise support vector machines (Kreßel, 1999),
3) error-correcting-output code (ECOC) support vector machines (Dietterich & Bakiri, 1995),
4) all-at-once support vector machines (Vapnik, 1998).

According to Vapnik's formulation, in one-against-all support vector machines, a k-class problem is converted into k two-class problems and for the ith two-class problem, class i is separated from the remaining classes. But by this formulation unclassifiable regions exist if we use the discrete decision functions. One way to solve this problem is to use continuous decision functions. Another is to introduce fuzzy membership functions as proposed in Inoue & Abe (2000).

In pairwise support vector machines, the k-class problem is converted into $k(k-1)/2$ two-class problems which cover all pairs of classes. But by this method also unclassifiable regions exist. Similar to Inoue & Abe (2000) unclassifiable regions can be resolved by introducing membership functions (Abe & Inoue, 2002).

In this paper, to resolve unclassifiable regions in LS-SVMs, we introduce fuzzy membership functions in the similar way as discussed in Abe & Inoue (2002), in which the membership function for a class is defined using the minimum operator. Here, we use the average operator as well as the minimum operator. By the minimum operator, the contour surface is parallel to the decision functions, thus the fuzzy SVM always outperforms the SVM. But this is not guaranteed for the average operator. In ECOC to resolve unclassifiable regions, the continuous Hamming distance is used. We can show that the average operator is complementary to the continuous Hamming distance.

Our purpose is to resolve the unclassifiable regions of LS-SVMs by the introduction of membership functions and then to clarify which operator is suited for LS-SVMs by computer experiments using several benchmark data sets.

This paper is organized as follows. In Section II, we describe the architectures of the SVM and the LS-SVM. In Section III, we discuss fuzzy LS-SVMs that resolve unclassifiable regions in multiclass problems. And in Section IV we show performance comparison of fuzzy SVMs and fuzzy LS-SVMs using some benchmark data sets.

II. Least Squares Support Vector Machines

In this section, we describe SVMs and LS-SVMs for two-class problems and the training method of LS-SVMs.

A. Architecture of SVMs

Let m-dimensional training data be $\mathbf{x}_i \, (i = 1, ..., M)$ and their class labels be y_i, where $y_i = 1$ and $y_i = -1$ for Classes 1 and 2, respectively. If these input data are linearly separable in the feature space, we can determine the following decision function:

$$D(\mathbf{x}) = \mathbf{w}^t \mathbf{g}(\mathbf{x}) + b, \tag{1}$$

where $\mathbf{g}(\mathbf{x})$ is a mapping function that maps $\mathbf{x}$ into the l-dimensional space, $\mathbf{w}$ is an l-dimensional vector and b is a scalar. To separate data linearly, the decision function satisfies the following condition:

$$y_i(\mathbf{w}^t \mathbf{g}(\mathbf{x}_i) + b) \geq 1 \quad \text{for} \quad i = 1, ..., M. \tag{2}$$

If the problem is linearly separable in the feature space, there are an infinite number of decision functions that satisfy (2). Among them we require that the hyperplane has the largest margin between two classes. Here the margin is the minimum distance from the separating hyperplane to the input data and this is given by $|D(\mathbf{x})|/||\mathbf{w}||$. And we call the separating hyperplane with the maximum margin optimal separating hyperplane.

Assuming that the margin is ρ, the following condition needs to be satisfied:

$$\frac{y_i D(\mathbf{x}_i)}{||\mathbf{w}||} \geq \rho \quad \text{for} \quad i = 1, ..., M. \tag{3}$$

We fix the product of ρ and $||\mathbf{w}||$:

$$\rho ||\mathbf{w}|| = 1. \tag{4}$$

Then, in order to obtain the optimal separating hyperplane with the maximum margin, we must find $\mathbf{w}$ with the minimum $||\mathbf{w}||$ that satisfy (3). From (4), this leads to solving the following optimization problem. Namely, minimize

$$\frac{1}{2}\mathbf{w}^t\mathbf{w} \tag{5}$$

subject to the constraints:

$$y_i(\mathbf{w}^t\mathbf{g}(\mathbf{x}_i) + b) \geq 1 \quad \text{for} \quad i = 1, ..., M. \tag{6}$$

When training data are not linearly separable, we introduce slack variables ξ_i (> 0) into (2) as follows:

$$y_i(\mathbf{w}^t\mathbf{g}(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for} \quad i = 1, ..., M. \tag{7}$$

The optimal separating hyperplane is determined so that the maximization of the margin and the minimization of the training error are achieved. Namely, minimize

$$\frac{1}{2}\mathbf{w}^t\mathbf{w} + \frac{C}{2}\sum_{i=1}^{n} \xi_i^p \tag{8}$$

subject to the constraints:

$$y_i(\mathbf{w}^t\mathbf{g}(\mathbf{x}_i) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \text{for} \quad i = 1, ..., M, \tag{9}$$

where C is a parameter that determines the tradeoff between the maximum margin and the minimum classification error and p is 1 or 2. When $p = 1$, the SVM is called L1 soft margin SVM (L1-SVM), and when $p = 2$, L2 soft margin SVM (L2-SVM). In the conventional SVM, optimal separating hyperplane is obtained by solving the above quadratic programming problem.

B. Architecture of LS-SVMs

In contrast to the SVM, the LS-SVM is trained by minimizing

$$\frac{1}{2}\mathbf{w}^t\mathbf{w} + \frac{C}{2}\sum_{i=1}^{M} \xi_i^2 \tag{10}$$

subject to the equality constraints:

$$y_i(\mathbf{w}^t\mathbf{g}(\mathbf{x}_i) + b) = 1 - \xi_i \quad \text{for} \quad i = 1, \cdots, M. \tag{11}$$

In the LS-SVM, we use equality constraints instead of inequality ones employed in the conventional SVM. Therefore, the optimal solution can be obtained by solving a set of linear equations instead of solving a quadratic programming problem.

To derive the dual problem of (10) and (11), we introduce the Lagrange multipliers as follows:

$$\begin{aligned} Q(\mathbf{w}, b, \alpha, \xi) \;&=\; \frac{1}{2}\mathbf{w}^t\mathbf{w} + \frac{C}{2}\sum_{i=1}^{M}\xi_i^2 \\ &\quad - \sum_{i=1}^{M} \alpha_i \{y_i(\mathbf{w}^t\mathbf{g}(\mathbf{x}_i) + b) - 1 + \xi_i\}, \end{aligned}$$

where $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_M)^t$ is Lagrange multipliers, which can be positive or negative in case of LS-SVM formulation. The conditions for optimality are derived by differentiating (12) with respect to $\mathbf{w}$, ξ_i, b, and α_i and equating the resulting equations to zero:

$$\mathbf{w} = \sum_{i=1}^{M} \alpha_i y_i \mathbf{g}(\mathbf{x}_i), \tag{12}$$

$$\sum_{i=1}^{M} \alpha_i y_i = 0, \tag{13}$$

$$\alpha_i = C\xi_i. \tag{14}$$

In a matrix form, (12), (13), (14), and (11) are expressed by

$$\begin{bmatrix} \boldsymbol{\Omega} & \mathbf{Y} \\ \mathbf{Y}^t & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ b \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ 0 \end{bmatrix}, \tag{15}$$

where $\boldsymbol{\Omega}$, $\mathbf{Y}$ and $\mathbf{1}$ are, respectively

$$\boldsymbol{\Omega}_{ij} = y_i y_j \mathbf{g}(\mathbf{x}_i)^t \mathbf{g}(\mathbf{x}_j) + \frac{\delta_{ij}}{C}, \tag{16}$$

$$\mathbf{Y} = (y_1, \cdots, y_M)^t, \tag{17}$$

$$\mathbf{1} = (1, \cdots, 1)^t. \tag{18}$$

Here,

$$\delta_{ij} = \begin{cases} 1 & i = j, \\ 0 & i \neq j. \end{cases}$$

C. Training of LS-SVMs

An LS-SVM is trained by solving (15). From (16), $\boldsymbol{\Omega}$ is positive definite. Thus,

$$\boldsymbol{\alpha} = \boldsymbol{\Omega}^{-1}(\mathbf{1} - \mathbf{Y}b). \tag{19}$$

Substituting (19) into the second matrix equation in (15), we obtain

$$b = \frac{\mathbf{Y}^t\boldsymbol{\Omega}^{-1}\mathbf{1}}{\mathbf{Y}^t\boldsymbol{\Omega}^{-1}\mathbf{Y}}. \tag{20}$$

Here, since $\boldsymbol{\Omega}$ is positive definite, $\boldsymbol{\Omega}^{-1}$ is also positive definite. In addition, since $\mathbf{Y}$ is non zero vector, $\mathbf{Y}^t\boldsymbol{\Omega}^{-1}\mathbf{Y} > 0$. Thus, b is always obtained.

Substituting (20) into (19), we obtain $\boldsymbol{\alpha}$. In solving (20), we use the Cholesky factorization.

Applying the Cholesky factorization to matrix $\boldsymbol{\Omega}$, it is decomposed into the product of two symmetric matrix as

$\mathbf{\Omega} = LL^t$. Here, L and L^t are lower and upper triangular matrixes, respectively, and each elements of L are given by

$$l_{op} = \left(q_{op} - \sum_{n=1}^{p-1} l_{pn} l_{on} \right) / l_{pp} \quad \text{for} \quad o = 1, ..., m, \quad p = 1, ..., o-1, \tag{21}$$

$$l_{aa} = \sqrt{q_{aa} - \sum_{n=1}^{a-1} l_{an}^2} \quad \text{for} \quad a = 1, ..., m. \tag{22}$$

Although $\mathbf{\Omega}$ is positive definite, the Cholesky factorization becomes unstable if $\mathbf{\Omega}$ is near positive semi-definite. Then, to prevent this, if

$$q_{aa} = \sum_{n=1}^{a-1} l_{an}^2 \leq \eta \tag{23}$$

where $\eta = (> 0)$, we set

$$l_{aa} = \sqrt{\eta}. \tag{24}$$

In this paper, we set η at 10^{-5}.

Then letting

$$\mathbf{\Omega a} = LL^t \mathbf{a} = \mathbf{Y}, \tag{25}$$

where $\mathbf{a}$ is an M-dimensional vector, we can solve (25) without matrix inversion. Using $\mathbf{a}$, b given by (20) reduces to

$$b = \frac{\mathbf{a}^t \mathbf{1}}{\mathbf{Y}^t \mathbf{a}}. \tag{26}$$

Similarly, we can obtain $\boldsymbol{\alpha}$.

D. Kernel Functions

One of the characteristic of the SVM is that it uses the technique called kernel trick. In (16), defining

$$K(\mathbf{x}, \mathbf{x}') = \mathbf{g}(\mathbf{x})^t \mathbf{g}(\mathbf{x}'), \tag{27}$$

where $K(\mathbf{x}, \mathbf{x}')$ is a kernel function, we can avoid treating variables in the feature space. In the following study, we use the kernel functions as follows:

- dot product kernels: $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^t \mathbf{x}'$;
- polynomial kernels: $K(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^t \mathbf{x}' + 1)^d$, where d is a positive integer;
- RBF kernels: $K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2)$, where γ is a positive parameter.

If the problem has a very large number of input variables, the value of a kernel function becomes so small or large that training of support vector machines becomes difficult.

For a polynomial kernel with degree d, the maximum value is $(m+1)^d$ if the range of input variables is $[0, 1]$. Thus, when m is very large, we normalize polynomial kernels as follows:

$$K(\mathbf{x}, \mathbf{x}') = \frac{(\mathbf{x}^t \mathbf{x}' + 1)^d}{(m+1)^d}. \tag{28}$$

Similarly, for RBF kernels, the maximum value of $||\mathbf{x} - \mathbf{x}||^2$ is m. Thus we normalize RBF kernels as follows:

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\frac{\gamma}{m} ||\mathbf{x} - \mathbf{x}'||^2). \tag{29}$$

III. Multiclass Classification

The formulation of SVMs is based on a two-class classification problem. Since SVMs determine the decision boundary directly, an extension to multiclass problem is not unique. Vapnik (1998) uses one-against-all classification, in which one class is separated from the remaining classes. By this formulation, however, we need to solve a set of linear equations with the number of variables equal to the number of training data. Therefore, long training time is necessary for training a problem with a large number of training data.

Considering the computational burden in training, here we use pairwise classification (Kreßel, 1999). In one-against-all and pairwise classifications, unclassifiable regions exist (Inoue & Abe, 2000; Abe & Inoue, 2002). Thus similar to Abe & Inoue (2002) we introduce fuzzy membership functions to resolve unclassifiable regions in pairwise classification.

In pairwise classification we require a binary classifier for each possible pair of classes and the number of the total pairs is $k(k-1)/2$ for a k-class problem. The decision function for the pair of classes i and j is given by

$$D_{ij}(\mathbf{x}) = \mathbf{w}_{ij}^t \mathbf{g}(\mathbf{x}) + b_{ij}, \tag{30}$$

where $D_{ij}(\mathbf{x}) = -D_{ji}(\mathbf{x})$. Then for the datum $\mathbf{x}$ we calculate

$$D_i(\mathbf{x}) = \sum_{j \neq i, i=1}^{k} \text{sign}(D_{ij}(\mathbf{x})), \tag{31}$$

where

$$\text{sign}(a) = \begin{cases} 1 & a > 0, \\ 0 & \text{otherwise}, \end{cases}$$

and this datum is classified into the class

$$\arg \max_{i=1,\ldots,k} D_i(\mathbf{x}). \tag{32}$$

If (32) is satisfied for one i, $\mathbf{x}$ is classified into class i. But if (32) is satisfied for plural i's, $\mathbf{x}$ is unclassifiable. In Fig.1, if $\mathbf{x}$ is in the shaded region, $D_i(\mathbf{x}) = 1$ for $i = 1, 2, 3$. Thus, from (32), $\mathbf{x}$ is unclassifiable. Therefore, the shaded region in the figure is unclassifiable.

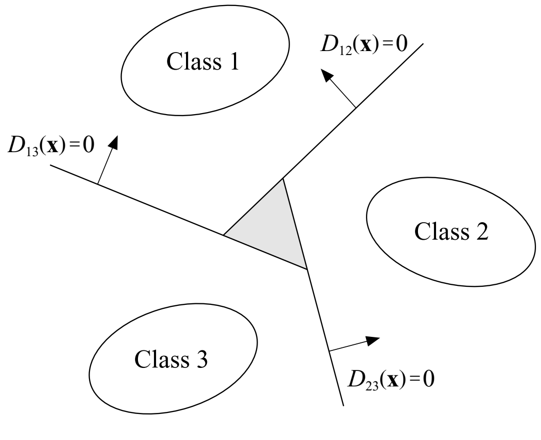

Fig. 1. An Unclassifiable Region by Pairwise Classification

To avoid this, similar to Abe & Inoue (2002), we introduce the fuzzy membership function. First, we define the one-dimensional membership function $m_{ij}(\mathbf{x})$ in the direction perpendicular to the optimal separating hyperplane $D_{ij}(\mathbf{x})$ as follows:

$$m_{ij} = \begin{cases} 1 & \text{for} \quad D_{ij}(\mathbf{x}) \geq 1, \\ D_{ij}(\mathbf{x}) & \text{otherwise.} \end{cases} \qquad (33)$$

Here, we allow negative degree of membership to make any data except those on the decision boundary be classified.

In Abe & Inoue (2002), the minimum operator is used for defining the membership function, $m_i(\mathbf{x})$, of $\mathbf{x}$ for class i. Here, we use the average operator as well as the minimum operator. The average operator is complementary to the continuous Hamming distance used in error correcting output codes (Dietterich & Bakiri, 1995).

Using the minimum operator the membership function, $m_i(\mathbf{x})$, of $\mathbf{x}$ for class i is given by

$$m_i(\mathbf{x}) = \min_{j=1,\ldots,k} m_{ij}(\mathbf{x}). \qquad (34)$$

Using the minimum operator, the shape of the membership function is a truncated polyhedral pyramid (Abe, 2001) in the feature space, and the contour surface, in which the degree of membership is the same, is parallel to the decision function as shown in Fig. 2.

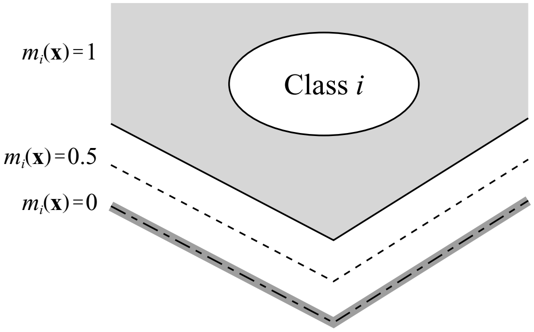

Fig. 2. Contour Lines of Membership Functions with Minimum Operator

Using the average operator the membership function, $m_i(\mathbf{x})$, of $\mathbf{x}$ for class i is given by

$$m_i(\mathbf{x}) = \frac{1}{k-1} \sum_{j \neq i, j=1}^{k} m_{ij}(\mathbf{x}). \qquad (35)$$

Using the average operator, the shape of the membership function is a truncated polyhedral pyramid but some part of the contour surface is not parallel to the decision function as shown in Fig.3.

Using either (34) or (35), the data $\mathbf{x}$ is classified into the class

$$\arg \max_{i=1,\ldots k} m_i(\mathbf{x}). \qquad (36)$$

Comparing the minimum and average operators, the regions where $m_i(\mathbf{x}) = 1$ are the same, but the regions where $m_i(\mathbf{x}) < a$ and $a < 1$, are different. We can show that the decision boundaries with the minimum operator are the same with those given by (31) for classifiable regions but the decision boundaries for the average operator are not. Thus the recognition rate using the minimum operator is always better than or equal to that by the conventional pairwise SVM. But this does not hold for the average operator.

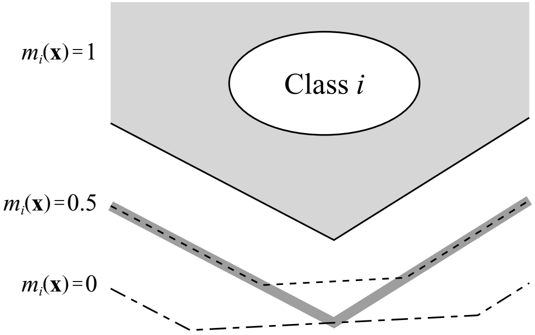

Fig. 3. Contour Lines of Membership Functions with Average Operator

Using the minimum operator, the unclassifiable region shown in Fig.1 is resolved as shown in Fig.4.

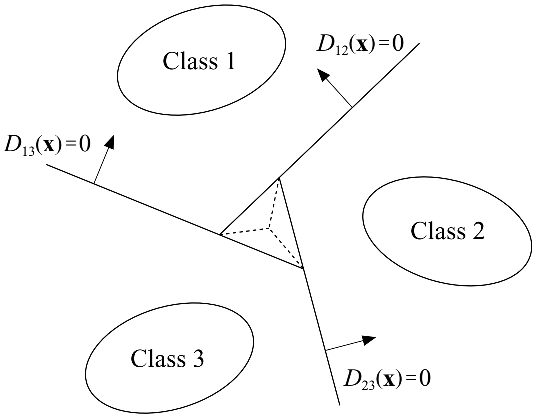

Fig. 4. Resolution of the Unclassifiable Region by the Minimum Operator

IV. Performance Evaluation

A. Condition of Experiments

Using the iris data (Fisher, 1936; Bezdek, Keller, Krishnapuram, Kuncheva, & Pal, 1999), the numeral data for license plate recognition (Takenaga, Abe, Takatoo, Kayama, Kitamura, & Okuyama, 1991), the thyroid data (Weiss & Kapouleas, 1999), the blood cell data (Hashizume, Motoike, & Yabe, 1988), and the hiragana data (Abe, 2001) listed in Table I, we compared the performance of the fuzzy LS-SVM (FLS-SVM) with minimum and average operators, the conventional LS-SVM, and the conventional L1 soft margin FSVM. The LS-SVM was used to show the improvement of the recognition rates by the FLS-SVM.

TABLE I

BENCHMARK DATA SPECIFICATION

Data	Inputs	Classes	Training data	Test data
Iris	4	3	75	75
Numeral	12	10	810	820
Thyroid	21	3	3772	3428
Blood cell	13	12	3097	3100
Hiragana-50	50	39	4610	4610
Hiragana-105	105	38	8375	8356
Hiragana-13	13	38	8375	8356

For the hiragana-105 and hiragana-50 data sets, the numbers of input valuables are over 50. Thus, we normalized the kernels. The value of the margin parameter C affects the recognition rates of the training and test data, but we evaluated the performance fixing C at 5000. We used the dot product, polynomial, and RBF kernels. The simulations were done on a Pentium III 1GHz PC.

B. Results and Discussions

Table II shows the results of the conventional FSVM, the LS-SVM, and the FLS-SVM with minimum and average operators. The highest recognition rates of the test data are shown in boldfaces. The FSVM performed best for 13 cases and the FLS-SVM with the minimum operator performed best for 16 cases. The FLS-SVM always outperforms the LS-SVM, which does not resolve unclassifiable regions. This is because the decision boundaries for the classifiable regions of the LS-SVM are the same with those of the FLS-SVM with the minimum operator. But this is not true for the FLS-SVM with the average operator. Performance of the FLS-SVM with the average operator is unstable; for some cases the performance is inferior to that of the LS-SVM.

C. Iris Data

The Fisher iris data are widely used for evaluating classification performance of classifiers. They consist of 150 data with four features and three classes; 50 data per class. We used the first 25 data of each class as the training data and the remaining 25 data of each class as the test data.

From Table II, the LS-SVM and the FLS-SVM show the same performance. This means that there are no data in the unclassifiable regions. The LS-SVM and the FLS-SVMs outperformed the FSVMs. But the FLS-SVMs with the average operator is inferior to the LS-SVM and the FLS-SVM.

D. Numeral Data

The numeral data were collected to identify Japanese license plates of running cars. They include numerals, hiragana and kanji characters. The original image taken from a TV camera was preprocessed and each numeral was transformed into 12 features such as the number of holes and the curvature of a numeral at some point.

In Table II, the recognition rates by the FSVM for $d = 3, 4$ are very low. This may be caused by a convergence problem since the recognition rates of the training data are low. The FSVM and the FLS-SVM with the minimum operator show comparable performance. For three cases out of four, the minimum operator performed better than the average operator did.

E. Thyroid Data

Thyroid data include 15 digital features and more than 92% of the data belong to one class. Thus the recognition rate smaller than 92% is useless.

The recognition rates of both training and test data by the FLS-SVM with minimum and average operators are much lower than those of the FSVM. The reason is not clear but this may be caused by the fact that almost all the input variables are discrete and that almost all the training data belong to one class. The results of the average operator are almost the same as those of the minimum operator.

F. Blood Cell Data

Blood cell classification involves classifying optically screened white blood cells into 12 classes using 13 features. This is a very difficult problem; class boundaries for some classes are ambiguous because the classes are defined according to the growth stages of white blood cells.

The FLS-SVM with minimum and average operators performed better than the FSVM on average and the FLS-SVM with the minimum operator performed better than the FLS-SVM with the average operator except for the polynomial kernel with degree 2.

G. Hiragana Data

Hiragana-50 and hiragana-105 data are gathered from Japanese license plates. The original gray-scale images of hiragana characters were transformed into 5×10-pixel and 7×15-pixel images, respectively with the gray-scale range being from 0 to 255. Then by performing gray-scale shift, position shift, and random noise addition to the images, the training and test data were generated. For the hiragana-105 data to reduce the number of input variables, i.e., $7 \times 15 = 105$, the hiragana-13 data were generated by calculating the 13 central moments (Cash & Hatamian, 1989). For the hiragana-105 and hiragana-50 data, we normalized the kernels.

For the hiragana-50, 105, and 13 data, the FLS-SVM with the minimum operator and the conventional FSVM show comparable performance except for dot product kernels. In case of the average operator, in most cases results are not good. Especially, they are very bad for dot product kernels.

H. Training Speed

Table III shows the training time of the FLS-SVM and conventional FSVM for the blood cell data. In training the FSVM, we used the primal-dual interior-point method with variable chunking. Starting from 50 training data, 50 data were added successively. In training the FLS-SVM, we used the Cholesky factorization method to solve the set of linear equations. The FLS-SVM requires 25 to 30 times longer training time than the FSVM. This is because the matrix $\mathbf{\Omega}$ shown in (15) becomes large if the number of input data become large. Therefore, as

TABLE II

PERFORMANCE OF FSVM AND FLS-SVM

Data	Kernel		FSVM		LS-SVM		FLS-SVM	(Min)	FLS-SVM	(Avg.)
			Test (%)	Train. (%)	Test (%)	Train. (%)	Test (%)	Train. (%)	Test (%)	Train. (%)
Iris	Dot		93.33	(100)	**97.33**	(100)	**97.33**	(100)	92.00	(100)
	Poly	$d = 2$	94.67	(100)	**98.67**	(100)	**98.67**	(100)	96.00	(100)
		$d = 3$	94.67	(100)	**96.00**	(100)	**96.00**	(100)	**96.00**	(100)
Numeral	Dot		**99.63**	(100)	99.27	(99.51)	99.39	(99.75)	99.15	(94.04)
	Poly	$d = 2$	**99.88**	(100)	97.56	(100)	98.17	(100)	98.54	(100)
		$d = 3$	89.88	(92.47)	98.17	(100)	**98.78**	(100)	98.66	(100)
		$d = 4$	89.63	(92.35)	98.42	(100)	**98.78**	(100)	98.66	(100)
Thyroid	Dot		**97.49**	(98.65)	93.35	(93.37)	93.79	(94.04)	93.73	(93.90)
	Poly	$d = 2$	**97.58**	(99.44)	94.75	(96.10)	95.10	(96.42)	95.27	(95.27)
		$d = 3$	**96.79**	(99.87)	93.20	(97.85)	93.85	(97.93)	93.70	(98.20)
		$d = 4$	**95.95**	(98.46)	90.67	(98.62)	92.12	(98.65)	92.01	(98.81)
	RBF	$\gamma = 0.5$	**97.41**	(98.57)	94.55	(95.84)	94.72	(95.56)	94.49	(95.33)
		$\gamma = 1$	**97.20**	(98.78)	94.55	(95.71)	95.94	(94.78)	94.75	(95.81)
Blood cell	Dot		90.90	(96.71)	92.48	(94.74)	**92.61**	(94.80)	88.32	(90.38)
	Poly	$d = 2$	92.19	(99.32)	93.52	(97.55)	**93.71**	(97.58)	92.72	(94.84)
		$d = 3$	91.97	(99.94)	92.16	(99.26)	**93.25**	(99.26)	93.13	(99.16)
		$d = 4$	**92.84**	(98.39)	90.13	(99.84)	92.10	(99.84)	92.16	(99.84)
	RBF	$\gamma = 0.5$	92.94	(97.32)	94.07	(96.48)	**94.10**	(96.51)	92.16	(94.51)
		$\gamma = 1$	92.97	(98.03)	93.84	(96.87)	**93.97**	(96.93)	92.84	(95.35)
Hiragana-50a	Dot		**98.31**	(100)	96.51	(99.96)	97.53	(99.96)	93.43	(98.48)
	Poly	$d = 2$	98.94	(100)	98.50	(100)	**99.11**	(100)	97.46	(100)
		$d = 3$	98.92	(100)	98.48	(100)	**99.15**	(100)	97.20	(100)
		$d = 4$	98.98	(100)	98.92	(100)	**99.07**	(100)	96.51	(100)
Hiragana-105	Dot		**99.93**	(100)	99.83	(99.90)	99.88	(100)	98.25	(98.97)
	Poly	$d = 2$	**100**	(100)	**100**	(100)	**100**	(100)	99.99	(100)
Hiragana-13	Dot		**99.35**	(99.95)	98.55	(99.45)	98.85	(99.57)	92.40	(96.17)
	Poly	$d = 2$	99.62	(100)	99.64	(99.96)	**99.70**	(99.96)	97.57	(99.98)
		$d = 3$	99.56	(100)	99.69	(100)	**99.79**	(100)	99.61	(100)

TABLE III

TRAINING TIME

Kernel		FSVM (sec)	FLS-SVM (sec)
Dot		3	86
Poly	$d = 2$	3	91
	$d = 3$	3	96
	$d = 4$	3	100
RBF	$\gamma = 0.5$	4	98
	$\gamma = 1$	4	98

indicated in Suykens & Vandewalle (1999), we need to use iterative methods for speedup.

I. Influence of the Outliers

Since LS-SVMs use equality constraints instead of inequality constraints, they are vulnerable to outliers (Suykens & Vandewalle, 1999). Only the difference between LS-SVMs and L2-SVMs is that the former uses equality constraints while the latter uses the inequality constraints. Thus, we compared their recognition performance when outliers were included.

For evaluation, we used the blood cell data belonging to Classes 2 and 3, which overlap heavily and are difficult to classify. As outliers, we added 10 data belonging to classes other than 2 and 3 to Class 2 training data. We used the polynomial kernel with degree 2.

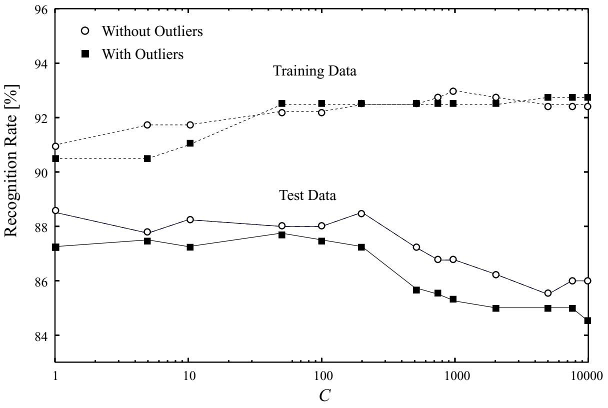

Fig. 5. Influence of Outliers to LS-SVM

Figures 5 and 6 show the recognition rates against the margin parameter C for the LS-SVM and L2-SVM, respectively. In the figures, the dotted lines show the recognition rates of

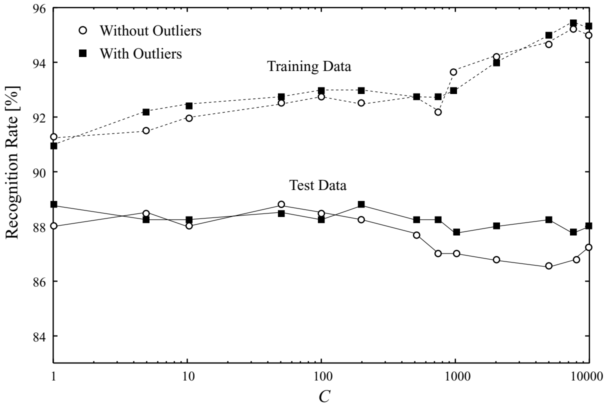

Fig. 6. Influence of Outliers to L2-SVM

the training data and the solid lines show those of test data. We calculated the recognition rate of the training data excluding outliers.

In Fig. 5, the recognition rates of the training data for the LS-SVM did not change much for $100 < C < 10000$ even if the outliers were included. But the recognition rate of the test data dropped rapidly, especially when outliers were included.

In Fig. 6, the recognition rates of the training data for the L2-SVM increased as the value of C was increased and there is no much difference between the recognition rates with and without outliers. In addition, the recognition rate of the test data with outliers was almost constant for the change of C and for a large value of C, it is better than that without outliers.

Comparing Figs. 5 and 6, we can see that the L2-SVM is more robust than the LS-SVM for outliers.

V. Conclusions

In this paper, we discussed fuzzy least squares support vector machines that resolve unclassifiable regions for multiclass problems. In defining membership functions for a class, we used minimum and average operators.

We evaluated the effectiveness of our method using several benchmark data sets. In most cases the fuzzy least squares support vector machine with the minimum operator showed comparable performance with the fuzzy support vector machine but the fuzzy least squares support vector machine with the average operator showed the worst performance.

References

Abe, S. (2001). *Pattern Classification, Neuro-fuzzy Methods and Their Comparison*. Springer-Verlag, London.

Abe, S. & Inoue, T. (2002) Fuzzy Support Vector Machines for Multiclass Problems. *Proceedings of the Tenth European Symposium on Artificial Neural Networks (ESANN 2002)*, 113–118.

Bezdek, J. C., Keller, J. M., Krishnapuram, R., Kuncheva, L. I., & Pal, N. H. (1999). Will the Real Iris Data Please Stand up? *IEEE Transactions on Fuzzy Systems*, 7 (3), 368–369.

Cash, G. L. & Hatamian, M. (1989). Optical Character Recognition by the Method of Moments. *Computer Vision, Graphics, and Image Processing*, 39, 291–310.

Dietterich, T. G. & Bakiri, G. (1995). Solving Multiclass Learning Problems via Error-Correcting Output Codes. *Journal of Artificial Intelligence Research*, 2, 263–286.

Fisher, R. A. (1936) The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7, 179–188.

Hashizume, A., Motoike, J., & Yabe, R. (1988). Fully Automated Blood Cell Differential System and Its Application, *Proceedings of IUPAC Third International Congress on Automation and New Technology in the Clinical Laboratory*, 297–302, Kobe, Japan.

Inoue, T. & Abe, S. (2000). Fuzzy Support Vector Machines for Pattern Classification. *Proceedings of International Joint Conference on Neural Networks (IJCNN'01)*, 1449–1454.

Kreßel, U. H.-G. (1999). Pairwise Classification and Support Vector Machines, In B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., *Advances in Kernel Methods: Support Vector Learning*, 255–268, MIT Press, Cambridge, MA.

Suykens, J. A. k. & Vandewalle, J. (1999). Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, 293–300.

Takenaga, H., Abe, S., Takatoo, M., Kayama, M., Kitamura, T., & Okuyama, Y. (1991). Input Layer Optimization of Neural Networks by Sensitivity Analysis and Its Application to Recognition of Numerals. *Electrical Engineering in Japan*, 111 (4), 130–138.

Vapnik, V. (1998). *Statistical Learning Theory*. John Wiley & Sons.

Weiss, S. M. & Kapouleas, I. (1999). An Empirical Comparison of Pattern Recognition, Neural Nets, and Machine Learning Classification Methods, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, Workshop ML3, 55–60.