



Comparison between error correcting output codes and fuzzy support vector machines

Kikuchi, Tomonori
Abe, Shigeo

(Citation)

Pattern Recognition Letters, 26(12):1937-1945

(Issue Date)

2005-09

(Resource Type)

journal article

(Version)

Accepted Manuscript

(URL)

<https://hdl.handle.net/20.500.14094/90000150>



Comparison between Error Correcting Output Codes and Fuzzy Support Vector Machines

Tomonori Kikuchi ^a, Shigeo Abe ^{*b}

^a Electrical and Electronics Engineering, Kobe University, Kobe, Japan

^b Graduate School of Science and Technology, Kobe University, Kobe, Japan

One-against-all support vector machines with discrete decision functions have unclassifiable regions. To resolve unclassifiable regions, support vector machines with continuous decision functions and fuzzy support vector machines have been proposed. If, in ECOC (error correcting output code) support vector machines, instead of discrete error functions, continuous error functions are used, unclassifiable regions are resolved. In this paper, first we prove that for one-against-all formulation, support vector machines with continuous decision functions are equivalent to fuzzy support vector machines with minimum and average operators. Then we discuss minimum operations as well as average operations for error functions of support vector machines and show the equivalence of ECOC support vector machines and fuzzy support vector machines for one-against-all formulation. Finally, we show by computer simulations that ECOC support vector machines are not always superior to one-against-all fuzzy support vector machines.

1. Introduction

Since support vector machines are formulated for two-class classification problems [1], an extension to multiclass problems is not unique. There are roughly four ways to solve this problem: one-against-all, pairwise, error-correcting output code, and all-at-once classifications.

Original formulation by Vapnik [1] is one-against-all classification, in which one class is separated from the remaining classes. By this formulation, however, unclassifiable regions exist. Instead of discrete decision functions, Vapnik [2, p. 438] proposed to use continuous decision functions. Namely, we classify a datum into the class with the maximum value of the decision functions. Inoue and Abe [3] proposed fuzzy support vector machines, in which membership functions are defined using the decision functions. Abe [4] showed that support vector machines with continuous decision functions and fuzzy support vector machines with minimum operators are equivalent.

In pairwise classification, the n -class problem

is converted into $n(n-1)/2$ two-class problems. Kreßel [5] showed that by this formulation, unclassifiable regions reduce, but still they remain. To resolve unclassifiable regions for pairwise classification, Platt, Cristianini, and Shawe-Taylor [6] proposed decision-tree-based pairwise classification called Decision Directed Acyclic Graph (DDAG). Pontil and Verri [7] proposed to use rules of a tennis tournament to resolve unclassified regions. Not knowing their work, Kijssirikul and Ussivakul [8] proposed the same method and called it Adaptive Directed Acyclic Graph (ADAG).

Classification by DDAGs or ADAGs is faster than pairwise fuzzy SVMs. But the problem is that the generalization ability depends on the structure of decision trees. To solve this problem for ADAGs, Phetkaew, Kijssirikul, and Rivepi-boon [9] proposed to reorder an ADAG so that the Euclidean norm sum of the coefficient vectors of the hyperplanes associated with the leaf nodes is minimized. Takahashi and Abe [10] proposed to optimize the structure so that the class pairs with higher generalization abilities are put in the upper nodes of the tree.

*E-mail: abe@eedept.kobe-u.ac.jp

URL: www2.kobe-u.ac.jp/~abe/index.html

Abe and Inoue [11] extended one-against-all fuzzy support vector machines to pairwise classification and Tsujinishi and Abe [12,13] proposed fuzzy least-squares support vector machines.

Dietterich and Bakiri [14] proposed error correcting output codes (ECOC) to enhance generalization ability of classifiers borrowing the idea of error correcting codes used for correcting bit errors in transmission channels. One-against-all formulation is a special case of error correcting codes with no error correcting capability, and by introducing “don’t” care bits, also is pairwise formulation [15]. Using the continuous Hamming distance for support vector machines, instead of the Hamming distance, unclassifiable regions are resolved.

In all-at-once formulation we need to determine all the decision functions at once [16,17], [2, pp. 437–440]. But this results in simultaneously solving a problem with a larger number of variables than the above mentioned methods.

In this paper, we discuss ECOC support vector machines in comparison to fuzzy support vector machines. First we summarize one-against-all fuzzy support vector machines and prove that fuzzy support vector machines with minimum and average operators are equivalent to support vector machines with continuous decision functions. Then we discuss ECOC support vector machines. First, we define the distance between codes by the maximum of continuous error functions as well as the continuous Hamming distance, which is the sum of continuous error functions. Next, we show that these definitions are equivalent to the fuzzy support vector machines with minimum and average operators, respectively. Then we compare recognition performance of ECOC support vector machines with that of one-against-all fuzzy support vector machines.

In Section 2, we explain two-class support vector machines, and in Section 3 we summarize one-against-all fuzzy support vector machines, and prove that support vector machines with continuous decision functions are equivalent to fuzzy support vector machines with minimum and average operators. Then in Section 4 we discuss ECOC support vector machines. In Section 5, we compare recognition performance of ECOC support

vector machines with average and minimum operators with one-against-all fuzzy support vector machines.

2. Two-class Support Vector Machines

Let M m -dimensional inputs \mathbf{x}_i ($i = 1, \dots, M$) belong to Class 1 or 2 and the associated labels be $y_i = 1$ for Class 1 and -1 for Class 2. Let the decision function be

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

where \mathbf{w} is an m -dimensional vector, b is a scalar, and

$$y_i D(\mathbf{x}_i) \geq 1 - \xi_i \quad \text{for } i = 1, \dots, M. \quad (2)$$

Here ξ_i are nonnegative slack variables.

The distance between the separating hyperplane $D(\mathbf{x}) = 0$ and the training datum, with $\xi_i = 0$, nearest to the hyperplane is called margin. The hyperplane $D(\mathbf{x}) = 0$ with the maximum margin is called optimal separating hyperplane.

To determine the optimal separating hyperplane, we minimize

$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^M \xi_i \quad (3)$$

subject to the constraints:

$$y_i (\mathbf{w}^t \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{for } i = 1, \dots, M, \quad (4)$$

where C is the margin parameter that determines the tradeoff between the maximization of the margin and minimization of the classification error. The data that satisfy the equality in (4) are called support vectors.

To enhance separability, the input space is mapped into the high-dimensional dot-product space called feature space. Let the mapping function be $\mathbf{g}(\mathbf{x})$. If the dot product in the feature space is expressed by $H(\mathbf{x}, \mathbf{x}') = \mathbf{g}^t(\mathbf{x})\mathbf{g}(\mathbf{x}')$, $H(\mathbf{x}, \mathbf{x}')$ is called kernel function, and we do not need to explicitly treat the feature space. The kernel functions used in this study are as follows:

1. Polynomial kernels

$$H(\mathbf{x}, \mathbf{x}') = (\mathbf{x}^t \mathbf{x}' + 1)^d, \quad (5)$$

where d is an integer.

2. RBF kernels

$$H(\mathbf{x}, \mathbf{x}') = \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2), \quad (6)$$

where γ is a positive parameter for slope control.

3. One-against-all Support Vector Machines

3.1. Support Vector Machines with Continuous Decision Functions

For the conventional support vector machine for an n -class classification problem, let the i th decision function that classifies class i and the remaining classes be

$$D_i(\mathbf{x}) = \mathbf{w}_i^t \mathbf{g}(\mathbf{x}) + b_i, \quad (7)$$

where $\mathbf{g}(\mathbf{x})$ is the mapping function that maps \mathbf{x} into the l -dimensional feature space, \mathbf{w}_i is the l -dimensional vector, and b_i is a scalar.

The hyperplane $D_i(\mathbf{x}) = 0$ forms the optimal separating hyperplane in the feature space and, if the training data are linearly separable, the support vectors belonging to class i satisfy $D_i(\mathbf{x}) = 1$ and those belonging to the remaining classes satisfy $D_i(\mathbf{x}) = -1$. For the conventional support vector machine, if for the input vector \mathbf{x}

$$D_i(\mathbf{x}) > 0 \quad (8)$$

satisfies for one i , \mathbf{x} is classified into class i . Since only the sign of the decision function is used, the decision is discrete.

If (8) is satisfied for plural i 's, or there is no i that satisfies (8), \mathbf{x} is unclassifiable. To avoid this, datum \mathbf{x} is classified into the class [2, p. 438]:

$$\arg \max_i D_i(\mathbf{x}). \quad (9)$$

Since the continuous value of the decision function determines classification, the decision is continuous.

3.2. Fuzzy Support Vector Machines

Here we summarize fuzzy support vector machines discussed in [3]. We introduce the membership function to resolve unclassifiable regions, while realizing the same classification results for

the data that satisfy (8) for one i . To do this, for class i we define one-dimensional membership functions $m_{ij}(\mathbf{x})$ in the directions orthogonal to the optimal separating hyperplanes $D_j(\mathbf{x}) = 0$ as follows:

1. For $i = j$

$$m_{ii}(\mathbf{x}) = \begin{cases} 1 & \text{for } D_i(\mathbf{x}) \geq 1, \\ D_i(\mathbf{x}) & \text{otherwise.} \end{cases} \quad (10)$$

2. For $i \neq j$

$$m_{ij}(\mathbf{x}) = \begin{cases} 1 & \text{for } D_j(\mathbf{x}) \leq -1, \\ -D_j(\mathbf{x}) & \text{otherwise.} \end{cases} \quad (11)$$

Since only the class i training data exist when $D_i(\mathbf{x}) \geq 1$, we assume that the degree of class i membership is 1, and otherwise, $D_i(\mathbf{x})$.

Here, we allow the negative degree of membership so that any data that are not on the boundary can be classified.

For $i \neq j$, class i is on the negative side of $D_j(\mathbf{x}) = 0$. In this case, support vectors may not include class i data but when $D_i(\mathbf{x}) \leq -1$, we assume that the degree of class i degree of membership is 1 and otherwise, $-D_j(\mathbf{x})$.

Using the minimum operator for $m_{ij}(\mathbf{x})$ ($j = 1, \dots, n$), we define the class i membership function of \mathbf{x} :

$$m_i(\mathbf{x}) = \min_{j=1, \dots, n} m_{ij}(\mathbf{x}). \quad (12)$$

The shape of the resulting membership function is the truncated polyhedral pyramid [18, pp. 76–78] in the feature space.

We can also define the class i membership function using the average operator:

$$m_i(\mathbf{x}) = \frac{1}{n} \sum_{j=1, \dots, n} m_{ij}(\mathbf{x}). \quad (13)$$

Now the datum \mathbf{x} is classified into the class

$$\arg \max_{i=1, \dots, n} m_i(\mathbf{x}). \quad (14)$$

Instead of using membership functions, it is possible to resolve unclassifiable regions by calculating Euclidean distances from the hyperplanes or posterior probabilities. But class regions by these methods do not necessarily include those determined by support vector machines with discrete decision functions.

3.3. Equivalence of FSVMs and SVMs with Continuous Decision Functions

It is shown in [4] that support vector machines with continuous decision functions are equivalent to fuzzy support vector machines with minimum operators for one-against-all classification. Here, we show that one-against-all support vector machines with continuous decision functions and one-against-all fuzzy support vector machines with average operators as well as minimum operators are equivalent in that they give the same classification result for any input.

Let $m_i^m(\mathbf{x})$ and $m_i^a(\mathbf{x})$ be the membership functions for class i using the minimum and average operators, respectively.

Then (12) and (13) are rewritten as follows:

$$\begin{aligned} m_i^m(\mathbf{x}) &= \min(\min(1, D_i(\mathbf{x})), \\ &\quad \min_{\substack{k=1, \dots, n \\ k \neq i}} \min(1, -D_k(\mathbf{x}))), \end{aligned} \quad (15)$$

$$\begin{aligned} m_i^a(\mathbf{x}) &= \frac{1}{n} (\min(1, D_i(\mathbf{x})) \\ &\quad + \sum_{k=1, k \neq i}^n \min(1, -D_k(\mathbf{x}))). \end{aligned} \quad (16)$$

Thus $m_i^a(\mathbf{x}) - m_j^a(\mathbf{x})$ is given by

$$\begin{aligned} m_i^a(\mathbf{x}) - m_j^a(\mathbf{x}) &= \frac{1}{n} [\min(1, D_i(\mathbf{x})) + \min(1, -D_j(\mathbf{x})) \\ &\quad - \min(1, D_j(\mathbf{x})) - \min(1, -D_i(\mathbf{x}))]. \end{aligned} \quad (17)$$

Now we prove the equivalence classifying the cases into three:

1. $D_i(\mathbf{x}) > 0, D_j(\mathbf{x}) \leq 0$ ($j = 1, \dots, n, j \neq i$)

By the support vector machine with continuous decision functions, input \mathbf{x} is classified into class i .

From (15) and the conditions on the signs of D_k ($k = 1, \dots, n$),

$$m_i^m(\mathbf{x}) > 0, \quad m_j^m(\mathbf{x}) \leq 0. \quad (18)$$

Thus by the fuzzy support vector machine with minimum operators, input \mathbf{x} is classified into class i .

From (17),

$$\begin{aligned} m_i^a(\mathbf{x}) - m_j^a(\mathbf{x}) &= \frac{1}{n} [\min(1, D_i(\mathbf{x})) + \min(1, -D_j(\mathbf{x})) \\ &\quad - D_j(\mathbf{x}) + D_i(\mathbf{x})] > 0. \end{aligned} \quad (19)$$

Thus by the fuzzy support vector machine with average operators, input \mathbf{x} is classified into class i .

2. $0 > D_i(\mathbf{x}) > D_j(\mathbf{x})$ ($j = 1, \dots, n, j \neq i$)

By the support vector machine with continuous decision functions, \mathbf{x} is classified into class i .

From (15) and the conditions on the signs of $D_k(\mathbf{x})$ ($k = 1, \dots, n$),

$$m_i^m(\mathbf{x}) > m_j^m(\mathbf{x}). \quad (20)$$

Thus input \mathbf{x} is classified into class i by the fuzzy support vector machine with minimum operators.

From (17),

$$\begin{aligned} m_i^a(\mathbf{x}) - m_j^a(\mathbf{x}) &= \frac{1}{n} [D_i(\mathbf{x}) - D_j(\mathbf{x}) - \min(1, -D_i(\mathbf{x})) \\ &\quad + \min(1, -D_j(\mathbf{x}))] > 0. \end{aligned} \quad (21)$$

Thus input \mathbf{x} is classified into class i by the fuzzy support vector machine with average operators.

3. $D_i(\mathbf{x}) > D_j(\mathbf{x}) > 0 > D_k(\mathbf{x})$ where $j \in N_1, k \in N_2, N_1 \cap N_2 = \emptyset, (N_1 \cup N_2) \cap \{i\} = \emptyset, N_1 \cup N_2 \cup \{i\} = \{1, \dots, n\}$

Input \mathbf{x} is classified into class i by the support vector machine with continuous decision functions.

From (15),

$$m_i^m(\mathbf{x}) = \min_{j \in N_1} -D_j(\mathbf{x}), \quad (22)$$

$$m_j^m(\mathbf{x}) = -D_i(\mathbf{x}) \quad \text{for } j \in N_1, \quad (23)$$

$$m_k^m(\mathbf{x}) = \min(-D_i(\mathbf{x}), D_k(\mathbf{x})) \\ \text{for } k \in N_2. \quad (24)$$

Thus,

$$m_i^m(\mathbf{x}) > m_j^m(\mathbf{x}) \quad \text{for } j \in N_1 \cup N_2. \quad (25)$$

Therefore, \mathbf{x} is classified into class i by the fuzzy support vector machine with minimum operators.

From

$$\begin{aligned} & m_i^a(\mathbf{x}) - m_j^a(\mathbf{x}) \\ &= \frac{1}{n} [\min(1, D_i(\mathbf{x})) - D_j(\mathbf{x})] \\ & \quad - \min(1, D_j(\mathbf{x}) + D_i(\mathbf{x})) > 0 \\ & \quad \text{for } j \in N_1 \end{aligned} \quad (26)$$

and from (19),

$$m_i^a(\mathbf{x}) > m_j^a(\mathbf{x}) \quad \text{for } j \in N_1 \cup N_2. \quad (27)$$

Thus, input \mathbf{x} is classified into class i by the fuzzy support vector machine with average operators.

Therefore, one-against-all support vector machines with continuous decision functions and the fuzzy support vector machines with minimum or average operators are equivalent.

4. Error-correcting Output Codes

Error correcting codes, which detect and correct errors in data transmission channels, are used to improve generalization ability in pattern classification. For support vector machines, in addition to generalization improvement they can be used to resolve unclassifiable regions. In this section, first we discuss how error-correcting codes can be used for pattern classification. Next, by introducing “don’t care” output, we discuss a unified scheme for output coding that includes one-against-all and pairwise formulations [15]. Then we show the equivalence of the error correcting codes with the membership functions.

Table 1

Error-correcting codes for three classes

Class	\mathbf{g}_1	\mathbf{g}_2	\mathbf{g}_3
1	1	-1	-1
2	-1	1	-1
3	-1	-1	1

4.1. Output Coding by Error-correcting Codes

Dietterich and Bakiri [14] proposed to use error-correcting output codes for multiclass problems. Let g_{ij} be the target value of the j th decision function $D_j(\mathbf{x})$ for class i :

$$g_{ij} = \begin{cases} 1 & \text{if } D_j(\mathbf{x}) > 0 \text{ for class } i, \\ -1 & \text{otherwise.} \end{cases} \quad (28)$$

The j th column vector $\mathbf{g}_j = (g_{1j}, \dots, g_{nj})^t$ is the target vector for the j th decision function, where n is the number of classes. If all the elements of a column are 1 or -1, classification is not performed by this decision function and two column vectors with $\mathbf{g}_i = -\mathbf{g}_j$ result in the same decision function. Thus the maximum number of distinct decision functions is $2^{n-1} - 1$.

The i th row vector (g_{i1}, \dots, g_{ik}) corresponds to a code word for class i , where k is the number of decision functions. In error-correcting codes, if the minimum Hamming distance between pairs of code words is h , the code can correct at least $\lfloor (h-1)/2 \rfloor$ -bit errors. For 3-class problems, there are three decision functions in maximum as shown in Table 1, which is equivalent to one-against-all formulation and there is no error-correcting function. Thus ECOC is considered to be a variant of one-against-all classification.

4.2. Unified Scheme for Output Coding

Introducing “don’t care” outputs, Allwein, Schapire, and Y. Singer [15] unified output codes that include one-against-all, pairwise, and ECOC schemes. Denoting a “don’t care” output by 0, pairwise classification [5] for three classes can be shown as in Table 2.

Table 2

Extended error-correcting codes for pairwise classification with three classes

Class	\mathbf{g}_1	\mathbf{g}_2	\mathbf{g}_3
1	1	0	-1
2	-1	1	0
3	0	-1	1

To calculate the distance of \mathbf{x} from the j th decision function for class i , we define the error $\varepsilon_{ij}(\mathbf{x})$ by

$$\varepsilon_{ij}(\mathbf{x}) = \begin{cases} 0 & \text{for } g_{ij} = 0, \\ \max(1 - g_{ij}D_j(\mathbf{x}), 0) & \text{otherwise.} \end{cases} \quad (29)$$

If $g_{ij} = 0$, we need to skip this case. Thus, $\varepsilon_{ij}(\mathbf{x}) = 0$. If $g_{ij}D_j(\mathbf{x}) \geq 1$, \mathbf{x} is on the correct side of the j th decision function with more than or equal to the maximum margin. Thus, $\varepsilon_{ij}(\mathbf{x}) = 0$. If $g_{ij}D_j(\mathbf{x}) < 1$, \mathbf{x} is on the wrong side or even if it is on the correct side, the margin is smaller than the maximum margin. We evaluate this disparity by $1 - g_{ij}D_j(\mathbf{x})$.

Then the distance of \mathbf{x} from class i is given by

$$d_i(\mathbf{x}) = \sum_{j=1}^k \varepsilon_{ij}(\mathbf{x}). \quad (30)$$

Using (30), \mathbf{x} is classified into

$$\arg \min_{i=1,\dots,n} d_i(\mathbf{x}). \quad (31)$$

Instead of (29), if we use the discrete function:

$$\varepsilon_{ij}(\mathbf{x}) = \begin{cases} 0 & \text{for } g_{ij} = 0, \\ 0 & \text{for } g_{ij} = \pm 1, g_{ij}D_j(\mathbf{x}) \geq 1, \\ 1 & \text{otherwise,} \end{cases} \quad (32)$$

(30) gives the Hamming distance. But by this formulation unclassifiable regions occur.

4.3. Equivalence of ECOC with Membership Functions

Here, we discuss the relationship between ECOC and membership functions. For $g_{ij} = \pm 1$,

the error $\varepsilon_{ij}(\mathbf{x})$ is expressed by the one-dimensional membership functions $m_{ij}(\mathbf{x})$:

$$\begin{aligned} m_{ij}(\mathbf{x}) &= \min(g_{ij}D_j(\mathbf{x}), 1) \\ &= 1 - \varepsilon_{ij}(\mathbf{x}). \end{aligned} \quad (33)$$

Thus, if we define the membership function for class i by

$$m_i(\mathbf{x}) = \frac{1}{\sum_{j=1}^k |g_{ij}|} \sum_{\substack{j=1 \\ g_{ij} \neq 0}}^k m_{ij}(\mathbf{x}) \quad (34)$$

and classify \mathbf{x} into

$$\arg \max_{i=1,\dots,n} m_i(\mathbf{x}), \quad (35)$$

we obtain the same recognition result as that by (31). This is equivalent to a fuzzy support vector machine with the average operator.

Similarly, instead of (30), if we use

$$d_i(\mathbf{x}) = \max_{j=1,\dots,n} \varepsilon_{ij}(\mathbf{x}), \quad (36)$$

the resulting classifier is equivalent to a fuzzy support vector machine with minimum operators.

5. Performance Evaluation

We evaluated recognition performance of ECOC support vector machines with one-against-all support vector machines using the blood cell data and hiragana data listed in Table 3. The blood cell classification involves classifying optically screened white blood cells into 12 classes [19]. 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. Hiragana data were gathered from Japanese license plates. The original gray-scale images of hiragana characters were transformed into 5×10 -pixel 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 [20]. Hiragana data are relatively easy to be classified.

As error correcting codes we used the BCH (Bose-Chaudhuri-Hochquenghem) codes, which

Table 3
Benchmark data specification

Data	Inputs	Classes	Train.	Test
Blood cell	13	12	3097	3100
Hiragana	50	39	4610	4610

belong to one type of cyclic codes. We used four BCH codes with 15, 31, 63, and 127 word lengths, properly setting the minimum Hamming distances. For each word length we trained 10 ECOC support vector machines with $C = 5000$ changing code words.

Table 4 shows the results for the blood cell data with polynomial kernels with degree 3. In the “Code” column, e.g., (15, 7) means that the word length is 15 bits and the minimum Hamming distance is 7. The “Hamming,” “Average,” and “Minimum” columns list the average recognition rate of the test and the training data (in the brackets) using the Hamming distance, the average operator, and the minimum operator, respectively. The boldfaced numeral shows the maximum recognition rate among different codes.

From the table, using the Hamming distance, the recognition rates of both training and test data improved as the word length was increased and they reached the maximum at the word length of 63. But since by the Hamming distance unclassifiable regions exist, the recognition rates are lower than by average and minimum operators. By the average and minimum operators, however, the one-against-all support vector machines showed the best recognition rates. This may be caused by the lower recognition rates of the training data for the ECOC support vector machines than by the one-against-all support vector machines.

Thus, to improve the recognition rate of the test data, we used the RBF kernels. Table 5 shows the results for the RBF kernels with $\gamma = 1$. The ECOC support vector machines showed better recognition performance than the one-against-all support vector machines. In addition, the average operator showed better recognition performance than the minimum operator.

Table 6 shows the results of hiragana data for the polynomial kernels with degree 3. The ECOC support vector machine with the average operator and the word length of 127 showed the best recognition performance. But some ECOC support vector machines showed lower recognition performance than the one-against-all support vector machine. Thus the performance of ECOC support vector machines was not stable.

Unlike blood cell data, since the recognition rates of the training data are near 100% using polynomial kernels, the improvement using RBF kernels was not recognized. Thus, we do not include the results here.

5.1. Discussions

For the blood cell data, ECOC support vector machines showed better performance than fuzzy support vector machines when RBF kernels were used, but for the hiragana data, improvement was not significant if any. This is because for the hiragana data fuzzy support vector machines could achieve relatively good performance and little was left for improvement.

For the blood cell data with polynomial kernels and for the hiragana data, ECOC support vector machines did not perform better than one-against-all support vector machines. And for the hiragana data, the recognition performance against the length of code word was unstable. Thus, to obtain good recognition performance, we need to optimize the structure of ECOC support vector machines.

As for the minimum operators and average operators, sometimes average operators performed better but there was not so much difference.

6. Conclusions

In this paper, first we proved that one-against-all support vector machines with continuous decision functions are equivalent to fuzzy support vector machines with minimum and average operators. Then, we discussed minimum operators as well as average operators for the error functions of ECOC support vector machines and showed the equivalence of ECOC support vector machines and fuzzy support vector machines for

Table 4

Recognition rates (%) of blood cell data with polynomial kernels ($d = 3$)

Code	Hamming	Average	Minimum
1-all	87.13 (92.41)	92.84 (96.09)	92.84 (96.09)
(15,7)	90.17 (93.34)	91.56 (94.45)	91.19 (93.95)
(31,11)	90.86 (93.60)	91.90 (94.59)	91.80 (94.16)
(63,31)	91.82 (94.64)	92.20 (94.98)	92.23 (94.32)
(127,63)	91.80 (94.58)	92.01 (94.82)	91.93 (96.09)

Table 5

Recognition rates (%) of blood cell data with RBF kernels ($\gamma = 1$)

Code	Hamming	Average	Minimum
1-all	86.68 (98.58)	92.94 (99.29)	92.94 (99.29)
(15,7)	92.43 (98.27)	93.47 (98.49)	93.07 (98.18)
(31,11)	92.88 (98.36)	93.85 (98.59)	93.53 (98.13)
(63,27)	93.68 (98.64)	94.05 (98.68)	93.75 (98.37)
(127,55)	93.68 (98.60)	93.96 (98.61)	93.63 (97.94)

one-against-all formulation. By computer simulations we showed that ECOC support vector machines are not always superior to one-against-all fuzzy support vector machines.

REFERENCES

1. V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, London, UK, 1995.
2. V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, NY, 1998.
3. T. Inoue and S. Abe. Fuzzy support vector machines for pattern classification. In *Proceedings of International Joint Conference on Neural Networks (IJCNN '01)*, volume 2, pages 1449–1454, July 2001.
4. S. Abe. Analysis of multiclass support vector machines. In *Proceedings of International Conference on Computational Intelligence for Modelling Control and Automation (CIMCA '2003)*, pages 385–396, Vienna, Austria, February 2003.
5. U. H.-G. Kreßel. Pairwise classification and support vector machines. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 255–268. The MIT Press, Cambridge, MA, 1999.
6. J. C. Platt, N. Cristianini, and J. Shawe-Taylor. Large margin DAGs for multiclass classification. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, pages 547–553. The MIT Press, Cambridge, MA, 2000.
7. M. Pontil and A. Verri. Support vector machines for 3-D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.
8. B. Kijsirikul and N. Ussivakul. Multiclass support vector machines using adaptive directed acyclic graph. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 2002)*, pages 980–985, 2002.

Table 6

Recognition rates (%) of hiragana data with polynomial kernels ($d = 3$)

Code	Hamming	Average	Minimum
1-all	97.55 (100)	99.28 (100)	99.28 (100)
(15,7)	95.50 (99.96)	97.63 (99.85)	96.93 (99.97)
(31,11)	98.38 (99.99)	99.01 (100)	98.56 (99.97)
(63,31)	99.01 (100)	99.30 (100)	99.17 (99.97)
(127,63)	99.31 (100)	99.46 (100)	99.26 (99.97)

9. T. Phetkaew, B. Kijirikul, and W. Rivepi-boon. Reordering adaptive directed acyclic graphs: An improved algorithm for multi-class support vector machines. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 2003)*, volume 2, pages 1605–1610, July 2003.
10. F. Takahashi and S. Abe. Optimizing directed acyclic graph support vector machines. In *Proceedings of Artificial Neural Networks in Pattern Recognition (ANNPR 2003)*, pages 166–170, September 2003.
11. S. Abe and T. Inoue. Fuzzy support vector machines for multiclass problems. In *Proceedings of the Tenth European Symposium on Artificial Neural Networks (ESANN'2002)*, pages 116–118, Bruges, Belgium, April 2002.
12. D. Tsujinishi and S. Abe. Fuzzy least squares support vector machines. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 2003)*, volume 2, pages 1559–1604, July 2003.
13. D. Tsujinishi and S. Abe. Fuzzy least squares support vector machines for multiclass problems. *Neural Networks*, 16(5–6):785–792.
14. T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.
15. E. L. Allwein, R. E. Schapire, and Y. Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.
16. K. P. Bennett. Combining support vector and mathematical programming methods for classification. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*, pages 307–326. The MIT Press, Cambridge, MA, 1999.
17. J. Weston and C. Watkins. Support vector machines for multi-class pattern recognition. In *Proceedings of the Seventh European Symposium on Artificial Neural Networks (ESANN'99)*, pages 219–224, 1999.
18. S. Abe. *Pattern Classification: Neuro-fuzzy Methods and Their Comparison*. Springer-Verlag, London, UK, 2001.
19. A. Hashizume, J. Motoike, and R. Yabe. Fully automated blood cell differential system and its application. In *Proceedings of the IUPAC Third International Congress on Automation and New Technology in the Clinical Laboratory*, pages 297–302, Kobe, Japan, September 1988.
20. M.-S. Lan, H. Takenaga, and S. Abe. Character recognition using fuzzy rules extracted from data. In *Proceedings of the Third IEEE International Conference on Fuzzy Systems*, volume 1, pages 415–420, Orlando, FL, June 1994.