Training of Kernel Fuzzy Classifiers by Dynamic Cluster Generation

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

Abstract

We discuss kernel fuzzy classifiers with hypersphere regions, which are defined in the feature space mapped from the input space. Starting from a hypersphere with a small radius defined at a datum, we expand the hypersphere if the new datum is within the prescribed distance from the center. And if not, we define a new hypersphere. After rule generation, we resolve overlaps of different classes contracting the hyperspheres. We then define a truncated conical membership function for each hypersphere. We demonstrate the usefulness of the kernel version of fuzzy classifiers with hypersphere regions with several benchmark data sets.

1 Introduction

In support vector machines (SVMs) [1], the input space is mapped into a high dimensional feature space, and in the space, the optimal hyperplane is determined so that two classes are separated with the maximum margin. According to performance comparisons in a wide range of applications, support vector machines have shown to have high generalization ability.

Inspired by the success of support vector machines, to improve generalization ability and classification ability, conventional pattern classification techniques are extended to incorporate maximizing margins and mapping to a feature space. For example, online perceptron algorithms, neural networks, and fuzzy systems have incorporated maximizing margins [2].

There are numerous conventional techniques that are extended to be used in the high-dimensional feature space, e.g., kernel perceptrons, the k-means clustering algorithm, the kernel self organizing feature map, kernel discriminant analysis, kernel principal component analysis, kernel Mahalanobis distance, and kernel least-squares [2].

One of the problems of support vector machines is that it is difficult to analyze the behavior of support vector machines because the input space is mapped to the high-dimensional feature space. There are several approaches to visualize support vector machines [3, 4, 5]. Another approach is to extend fuzzy classifiers to kernel fuzzy classifiers that are defined in the feature space [6]. Although fuzzy classifiers with hyperbox regions such as discussed in [7, 8] are difficult to extend to the feature space, classifiers with hyperspheres [9, 10, 11] are relatively easily extended.

In this paper, we define fuzzy rules in the feature space in the way similar to fuzzy min-max classifiers [7]. Instead of generating and contracting hyperboxes, we generate and contract hyperspheres in the feature space. To facilitate efficient calculations in the feature space, we use kernel tricks. Namely, we use kernel functions associated with a mapping function to avoid explicit treatment of variables in the feature space.

Suppose there are training data belonging to one of n classes. We scan the training data and for a training datum with no associated hyperspheres we define the hypersphere at the training datum with a predefined small radius R_ε. If there is a hypersphere that includes the training datum or whose center is within the prescribed maximum radius $R_{\max}$, we modify the center and the radius of the hypersphere. If not, we generate the hypersphere at the training datum with radius R_ε. After generating the hyperspheres, we check whether the hyperspheres of different classes overlap. If there is an overlap, we resolve the overlap contracting the hyperspheres.

In Section 2, we discuss the rule generation of kernel fuzzy classifiers. In Section 3, we compare the generalization performance of the method with that of other classifiers using two-class and multiclass data sets.

2 Dynamic Rule Generation

2.1 Concept

In the first stage we scan the training data and generate hyperspheres of each class without resolving overlaps between classes. Then in the second stage, we resolve overlaps between classes, contracting hyperspheres.

We explain the idea of hypersphere generation using the two-dimensional example shown in Fig. 1. In the figure, assume that Data 1, 2, and 3 belonging to the same class are scanned in this order. For Datum 1, we generate the circle with Datum 1 being the center and with radius R_ε. Then for Datum 2, we check if the circle is expandable. Assume that distance between Data 1 and 2 is shorter than $R_{\max}$. Then we generate the dotted circle shown in Figure 1, whose center is at the middle of data 1 and 2 and whose diameter is the distance between Data 1 and 2.

If Datum 3 is inside of the dotted circle, we update the center and the radius. But because Datum 3 is outside of the circle, we check if the distance between the center and the datum is within $R_{\max}$. If so, we update the center adding Datum 3 in addition to Datum 1 and 2 and calculate the minimum radius that includes Data 1, 2, and 3. If the circle is not expandable, we generate the circle with Datum 3 being the center and with radius R_ε.

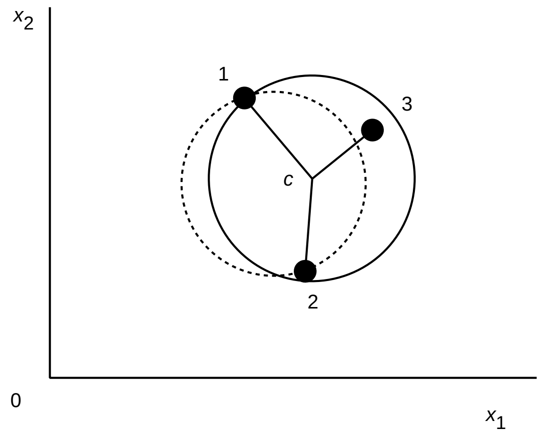

Figure 1. Generation of circles

After generating hyperspheres for each class without considering the overlap between classes, we resolve overlaps by contracting hyperspheres. In Fig. 2, two circles belonging to classes i and j are overlapped. We resolve this overlap, contracting the circles as shown in the figure.

For each hypersphere we define a membership function for datum $\mathbf{x}$, in which the degree of membership is 1 if $\mathbf{x}$ is in the sphere and the degree of membership decreases as $\mathbf{x}$ moves away from the hypersphere. Fig. 3 shows an example for a two dimensional case. The shape of the membership function is a truncated conical. Usually, the degree of membership is defined between 0 and 1, but to avoid unclassifiable regions, we assume negative degree of membership.

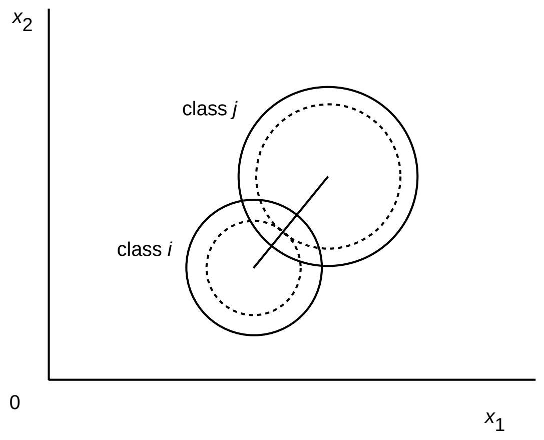

Figure 2. Resolution of overlap

2.2 Rule Generation

In the following we discuss the procedure of rule generation in detail.

Let the training inputs for class i be $\mathbf{x}_{ij}$ for $j = 1, \dots, M_i$, where $\mathbf{x}_{ij}$ is the jth training datum for class i and M_i is the number of data for class i. And let $\mathbf{g}(\mathbf{x})$ be the mapping function that maps the input space into the feature space.

The procedure for generating hyperspheres for class i is as follows.

1. Generate the hypersphere S_{i1} with center $\mathbf{c}_{i1} = \mathbf{g}(\mathbf{x}_{i1})$ and with radius $R_{i1} = R_\varepsilon$. Set $N_i^{\mathrm{s}} = 1$, $X_{i1} = \{1\}$, and $j = 2$, where N_i^{s} is the number of generated hyperspheres for class i and X_{i1} is the set of indices of data for calculating the center $\mathbf{c}_{i1}$.

2. Check if $\mathbf{x}_{ij}$ is in a sphere. Namely if there exists such k ($1 \le k \le N_i^{\mathrm{s}}$) that satisfies

$$\|\mathbf{g}(\mathbf{x}_{ij}) - \mathbf{c}_{ik}\| \le R_{ik}, \tag{1}$$

$\mathbf{x}_{ij}$ is in hypersphere S_{ik}, where

$$\mathbf{c}_{ik} = \frac{1}{|X_{ik}|} \sum_{j' \in X_{ik}} \mathbf{g}(\mathbf{x}_{ij'}). \tag{2}$$

Here, $|X_{ik}|$ is the number of elements in X_{ik}. If there are more than one hypersphere that satisfy (1), we select one whose value on the left-hand side of (1) is the smallest. Update the center given by (2) adding $\mathbf{x}_{ij}$, update R_{ik}, and go to Step 4. Otherwise, find hypersphere S_{ik} whose center is nearest to $\mathbf{x}_{ij}$:

$$k = \arg\min_{k'} \|\mathbf{g}(\mathbf{x}_{ij}) - \mathbf{c}_{ik'}\|. \tag{3}$$

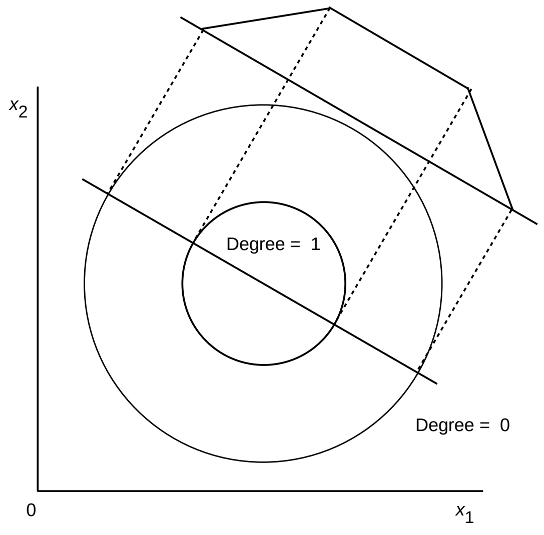

Figure 3. Definition of a membership function

3. Check if the hypersphere S_{ik} can be expandable for the inclusion of $\mathbf{x}_{ij}$. If

$$\|\mathbf{g}(\mathbf{x}_{ij'}) - \mathbf{c}_{ik}\| \leq R_{\max} \quad \text{for } j \in X_{ik} \tag{4}$$

is satisfied, hypersphere S_{ik} is expandable.

Then, we set

$$X_{ik} \leftarrow X_{ik} \cup \{\mathbf{x}_{ij}\}, \tag{5}$$
$$R_{ik} = \max_{j' \in X_{ik}} \|\mathbf{x}_{ij'} - \mathbf{c}_{ij}\|. \tag{6}$$

Otherwise, we set

$$N_i^{\mathrm{s}} \leftarrow N_i^{\mathrm{s}} + 1, \tag{7}$$
$$X_{i,N_i^{\mathrm{s}}} = \{j\}, \tag{8}$$

and generate the hypersphere $S_{i,N_i^{\mathrm{s}}}$ with center $\mathbf{c}_{i,N_i^{\mathrm{s}}} = \mathbf{g}(\mathbf{x}_{ij})$ and with radius $R_{i,N_i^{\mathrm{s}}} = R_\varepsilon$.

4. If $j = M_i$, terminate the algorithm. Otherwise, $j \leftarrow j + 1$ and go to Step 2.

2.3 Overlap Resolution

We can resolve overlaps by contracting hyperspheres or tuning membership functions, which are defined in the next section. But since we are going to define truncated conical membership functions shown in Fig. 3, we need to resolve overlaps between hyperspheres.

The ratio of contraction directly influences the generalization ability. But here, we resolve overlaps of hyperspheres S_{ij} and S_{ko} $(i \neq k)$ in the following way:

1. For $R_{ij} \leq \|\mathbf{c}_{ij} - \mathbf{c}_{ko}\| < R_{ij} + R_{ko}$ and $R_{ko} \leq \|\mathbf{c}_{ij} - \mathbf{c}_{ko}\| < R_{ij} + R_{ko}$

   As shown in Fig. 2, two hyperspheres overlap but each center is outside of the other hypersphere. We resolve the overlap contracting the hyperspheres by

$$\Delta r = \frac{R_{ij} + R_{ko} - \|\mathbf{c}_{ij} - \mathbf{c}_{ko}\|}{2}, \tag{9}$$
$$R_{ij} \leftarrow R_{ij} - \Delta r, \tag{10}$$
$$R_{ko} \leftarrow R_{ko} - \Delta r. \tag{11}$$

2. For $R_{ij} < \|\mathbf{c}_{ij} - \mathbf{c}_{ko}\| \leq R_{ko}$

   One of the hypersphere centers is inside of the other hypersphere but the other is not. We resolve the overlap contracting the hyperspheres by

$$R_{ko} \leftarrow \|\mathbf{c}_{ij} - \mathbf{c}_{ko}\| - \frac{R_{ij}}{2}, \tag{12}$$
$$R_{ij} \leftarrow \frac{R_{ij}}{2}. \tag{13}$$

3. For $\|\mathbf{c}_{ij} - \mathbf{c}_{ko}\| < R_{ij} \leq R_{ko}$

   Both centers of the hyperspheres are inside of the other hyperspheres. We resolve the overlap contracting the hyperspheres by

$$R_{ij} \leftarrow \frac{\|\mathbf{c}_{ij} - \mathbf{c}_{ko}\|}{2}, \tag{14}$$
$$R_{ko} \leftarrow \frac{\|\mathbf{c}_{ij} - \mathbf{c}_{ko}\|}{2}. \tag{15}$$

Contraction of hyperspheres may result in misclassification of the correctly classified training data, and thus may worsen the generalization ability. This is avoided by tuning membership functions. But here, we do not consider tuning. Instead, we optimize the value of $R_{\max}$ by cross validation.

2.4 Definition of Membership Functions

We define the membership function of S_{ij} for $\mathbf{x}$, $m_{ij}(\mathbf{x})$, by

$$m_{ij}(\mathbf{x}) = \begin{cases} 1 & \text{for} \quad d_{ij}(\mathbf{x}) \leq R_{ij}, \\ 1 - d_{ij}(\mathbf{x}) + R_{ij} & \text{for} \quad d_{ij}(\mathbf{x}) > R_{ij}, \end{cases} \tag{16}$$

where $d_{ij}(\mathbf{x})$ is the distance between $\mathbf{g}(\mathbf{x})$ and $\mathbf{c}_{ij}$ and is given by

$$d_{ij}(\mathbf{x}) = \|\mathbf{c}_{ij} - \mathbf{g}(\mathbf{x})\|. \tag{17}$$

The membership function given by (16) takes negative value so that any data are classified into a definite class unless they are on the class boundaries. Since the slopes of the membership functions are the same for all the hyperspheres, the class boundary on the line segment between $\mathbf{c}_{ij}$ and $\mathbf{c}_{ko}$

is the middle point of the line segment between the two hyperspheres S_{ij} and S_{ko}. This is a similar idea of the optimal separating hyperplane of a support vector machine. But if there are misclassified data belonging to class i or j, we can tune the slopes of membership functions [12]. But here we do not consider this.

2.5 Kernel Methods

In kernel methods, we treat the variables in the feature space implicitly using kernels. In our proposed method, we need to calculate $\|\mathbf{c}_{ij} - \mathbf{c}_{ko}\|$ and $\|\mathbf{g}(\mathbf{x}) - \mathbf{c}_{ij}\|$ using $H(\mathbf{x}, \mathbf{x}') = \mathbf{g}^T(\mathbf{x})\, \mathbf{g}(\mathbf{x})$. Namely,

$$\begin{aligned}
\|\mathbf{c}_{ij} - \mathbf{c}_{ko}\|^2 &= (\mathbf{c}_{ij} - \mathbf{c}_{ko})^T (\mathbf{c}_{ij} - \mathbf{c}_{ko}) \\
&= \frac{1}{|X_{ij}|^2} \sum_{\mathbf{x},\mathbf{x}' \in X_{ij}} H(\mathbf{x}, \mathbf{x}') \\
&\quad - \frac{2}{|X_{ij}|\,|X_{ko}|} \sum_{\mathbf{x} \in X_{ij}, \mathbf{x}' \in X_{ko}} H(\mathbf{x}, \mathbf{x}') \\
&\quad + \frac{1}{|X_{ko}|^2} \sum_{\mathbf{x},\mathbf{x}' \in X_{ko}} H(\mathbf{x}, \mathbf{x}'), \qquad (18)
\end{aligned}$$

$$\begin{aligned}
\|\mathbf{g}(\mathbf{x}) - \mathbf{c}_{ij}\|^2 &= H(\mathbf{x}, \mathbf{x}) - \frac{2}{|X_{ij}|} \sum_{\mathbf{x}' \in X_{ij}} H(\mathbf{x}, \mathbf{x}') \\
&\quad + \frac{1}{|X_{ij}|^2} \sum_{\mathbf{x}',\mathbf{x}'' \in X_{ij}} H(\mathbf{x}', \mathbf{x}''). \qquad (19)
\end{aligned}$$

In our study, we use the following kernels:

1. linear kernels: $H(\mathbf{x}, \mathbf{x}') = \mathbf{x}^t \mathbf{x}'$,
2. polynomial kernels:

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

   where d is a positive integer,
3. RBF kernels:

   $$H(\mathbf{x}, \mathbf{x}') = \exp(-\gamma ||\mathbf{x} - \mathbf{x}'||^2), \qquad (20)$$

   where γ is a positive parameter.
4. Mahalanobis kernels:

   $$H(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\delta}{m} (\mathbf{x} - \mathbf{x}')^T Q^{-1} (\mathbf{x} - \mathbf{x}')\right),$$

   where δ is a positive parameter and Q is the diagonal covariance matrix calculated using the training data [13].

Because, except for linear kernels,

$$\sum_i \mathbf{g}(\mathbf{x}_i) \neq \mathbf{g}(\sum_i \mathbf{x}_i) \qquad (21)$$

holds, we need to calculate the second term in the right hand side of (19) for $\mathbf{x}$ but the result of the third term may be saved to reduce computation time. Likewise, by saving the result of (18), computation time is reduced.

3 Performance Evaluation

We compared the generalization ability of the kernel fuzzy classifier and other methods using two groups of data sets: (1) some of the two-class data sets used in [14, 15][1] and (2) multiclass data sets used in [2, 12]. We used the linear and Mahalanobis kernels for the kernel fuzzy classifier.

Throughout the experiments we set $R_\varepsilon = 0.01$ and selected the value of $R_{\max}$ from 0.05 to 1.5 by the increment of 0.05 by 10-fold cross validation. In addition to this we set a large value to $R_{\max}$ to avoid clustering the class data. For Mahalanobis kernels we need to determine $R_{\max}$ and δ. Like Mahalanobis kernels for support vector machines [13], we determine the values of $R_{\max}$ and δ by line search. Namely, first we determine the value of $R_{\max}$ with $\delta = 1$ by cross validation. Then with the determined value of $R_{\max}$, we determine the value of δ by cross validation changing δ from 0.1 to 2 with the increment of 0.1.

3.1 Evaluation for Two-class Problems

Table 1 lists the specifications of two-class problems. Each problem has 100 or 20 training data sets and their corresponding test data sets. We determined the optimal values of $R_{\max}$ by 10-fold cross validation. For comparison, we used the support vector machine (SVM) with Mahalanobis kernels and the kernel fuzzy classifier with ellipsoidal regions [6]. For the support vector machine, we determined the optimum value of the margin parameter C and δ by line search. For the kernel fuzzy classifier with ellipsoidal regions (KFC-ER), we used RBF kernels and determined the parameters by 5-fold cross validation.

We performed cross validation of the first five training data sets, and selected the median of the best values. Then, for the optimal values, we trained the classifier for 100 or 20 training data sets and calculated the average recognition error and the standard deviation for the test data sets.

Table 2 lists the parameter values for the kernel fuzzy classifiers determined by cross validation. The symbol ∞ denotes that the highest recognition rate for the validation data set was obtained when the class data were not clustered. Except for the banana and image data sets relatively large values were selected for $R_{\max}$.

Table 3 lists the average classification errors and the standard deviations with the $\pm$ symbol. The "KFC-L" and

[1]http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm

Table 3. Average error rates and standard deviations of the test data sets.

Data	KFC-L	KFC	KFC-ER	SVM
Banana	12.6 ± 0.9	12.4 ± 0.8	10.9 ± 0.6	**10.4**$\pm \mathbf{0.5}$
B. Cancer	31.6 ± 5.3	31.0 ± 4.6	26.5 ± 4.4	**25.6**± 4.4
Diabetes	33.2 ± 5.9	28.7 ± 2.5	25.3± 2.0	**23.7**$\pm \mathbf{1.7}$
German	26.5 ± 2.4	27.4 ± 2.2	25.2± 2.4	**23.9**$\pm \mathbf{2.1}$
Heart	18.7 ± 3.5	16.2 ± 3.3	**15.6**± 3.6	15.7 ± 3.2
Image	**2.8**± 0.7	3.7 ± 0.5	2.9± 0.7	3.0± 0.5
Ringnorm	24.4± 0.6	**1.4**± 0.1	3.2 ± 0.3	1.7± 0.1

Table 1. Benchmark data sets for two-class problems.

Data	Inputs	Train.	Test	Sets
Banana	2	400	4900	100
B. cancer	9	200	77	100
Diabetes	8	468	300	100
German	20	700	300	100
Heart	13	170	100	100
Image	18	1300	1010	20
Ringnorm	20	400	7000	100

Table 2. Parameter values selected by cross validation.

Data	Linear	Mahalanobis	
	$R_{\max}$	$R_{\max}$	δ
Banana	0.1	0.35	0.7
B. Cancer	0.8	0.9	1.0
Diabetes	0.65	0.6	0.5
German	0.8	0.5	0.5
Heart	∞	∞	0.1
Image	0.05	0.1	1.4
Ringnorm	∞	∞	1.2

"KFC" columns list the values for the kernel fuzzy classifier with linear kernels and that with Mahalanobis kernels, respectively.

The best performance in the row is shown in boldface. For the german and image data sets, the KFC-L performed better than the KFC, but for the other four data sets, the KFC performed better.

The KFC-L showed the best performance for the image data set, but for the ringnorm data set, the KFC-L showed a large error compared to other methods. Table 4 shows the best recognition rates of the ringnorm validation data sets. The recognition rates of the KFC-L are very low compared to those of the KFC. Because smaller $R_{\max}$ results in overfitting, the large value of $R_{\max}$ was selected as optimal. And the large value caused large contraction of hyperspheres. Thus, in such a situation we need to optimize the membership functions as discussed in [12]. Except for the heart, image, and ringnorm data sets, KFC or KFC-L performed poorly compared to KFC-ER and SVM. For these data sets to improve the generalization ability we need to optimize the membership functions.

3.2 Evaluation for Multiclass Problems

As multiclass data sets, we used the data sets in [2, 12]. They were the iris data [16, 17], the numeral data for license plate recognition [18], the thyroid data [19],[2] the blood cell data [20], and hiragana data [12, 21]. Table 5 lists the specifications of the data sets.

We used pairwise support vector machines. To resolve unclassifiable regions, we used fuzzy support vector machines with minimum operators [2].

Table 6 shows the parameter values determined by cross validation. Unlike two-class problems, a relatively small value was set to $R_{\max}$.

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

Table 4. Cross validation results (%) for the ringnorm data set.

Data	KFC-L		KFC		
	$R_{\max}$	Rec.	$R_{\max}$	δ	Rec.
1	∞	75.75	∞	1.0	99.00
2	∞	76.75	∞	1.6	98.75
3	∞	74.00	∞	1.2	98.75
4	1.2	77.50	∞	1.1	99.00
5	∞	75.00	∞	1.4	98.75

Table 5. Benchmark data sets for multi-class problems.

Data	Inputs	Classes	Train.	Test
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

Table 6. Parameter values selected by cross validation.

Data	Linear	Mahalanobis	
	$R_{\max}$	$R_{\max}$	δ
Iris	0.3	0.35	0.6
Numeral	0.05	0.65	0.9
Thyroid	0.1	0.25	0.6
Blood cell	0.1	0.05	0.2
Hiragana-50	0.05	0.15	1.0
Hiragana-13	0.05	0.05	0.3
Hiragana-105	0.05	0.10	0.5

Table 7 lists the recognition rates of the test data sets for 6 classifiers. The results of the fuzzy mini-max classifier (FMM) and the k-nearest neighbor classifier (k-NN) are from [12]. They showed the best performance for the test data. For k-NN, the best k was selected from 1, 3, 5, and 7. But for other classifiers, the parameters were determined by cross validation. The SVM used Mahalanobis kernels and the margin parameter and δ were determined by cross validation. For the KFC-ER, polynomial kernels were used and the polynomial degree was determined by 5-fold cross validation.

The performance difference between the KFC-L and KFC is small. The reason may be that because the multi-class data sets are relatively easily classified compared to the two-class problems, the improvement of separability by mapping to the feature space resulted in overfitting. For the thyroid data, the KFC-L and KFC show poor recognition rates. This is the same tendency with that of the least-squares support vector machines [2]. But for other data sets, performance is comparable.

4 Conclusions

In this paper we discussed kernel fuzzy classifiers with hypersphere regions. We scan the training data and if no hyperspheres are defined for the class associated with a datum we define the hypersphere at the training datum with a predefined small radius. If there is a hypersphere that can include the training data within the prescribed maximum radius, we expand the hypersphere. If not, we generate the hypersphere at the training data with the small radius. After generating the hyperspheres, we resolve the overlap contracting the hyperspheres.

For some two-class data sets the proposed classifiers showed performance inferior to support vector machines and kernel fuzzy classifiers with ellipsoidal regions, but for multiclass problems, their performance was comparable to other methods.

To further improve the generalization ability, we need to tune the membership functions.

References

[1] V. N. Vapnik. *Statistical Learning Theory*. John Wiley & Sons, New York, 1998.

[2] S. Abe. *Support Vector Machines for Pattern Classification*. Springer-Verlag, London, 2005.

[3] D. Caragea, D. Cook, and V. Honavar. Towards simple, easy-to-understand, yet accurate classifiers. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM 2003)*, pages 497–500, Melbourne, FL, 2003.

Table 7. Recognition rates of the test data sets.

Data	KFC-L	KFC	FMM	KFC-ER	k-NN	SVM
Iris	**97.33**	96.00	**97.33**	**97.33**	94.67	96.00
Numeral	99.27	99.02	**99.76**	99.51	99.51	**99.76**
Thyroid	92.82	93.76	**99.42**	96.82	93.67	98.02
Blood cell	90.84	90.90	90.45	**93.16**	90.13	92.74
Hiragana-50	96.75	96.98	94.64	97.61	97.16	**98.52**
Hiragana-13	99.52	99.58	98.22	**99.86**	99.55	99.83
Hiragana-105	99.99	99.99	99.44	**100**	99.99	**100**

[4] X. Fu, C.-J. Ong, S. Keerthi, G. G. Hung, and L. Goh. Extracting the knowledge embedded in support vector machines. In *Proceedings of International Joint Conference on Neural Networks (IJCNN 2004)*, volume 1, pages 291–296, Budapest, Hungary, 2004.

[5] H. Núñẽz, C. Angulo, and Català. Rule extraction from support vector machines. In *Proceedings of the Tenth European Symposium on Artificial Neural Networks (ESANN 2002)*, pages 107–112, Bruges, Belgium, 2002.

[6] 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.

[7] P. K. Simpson. Fuzzy min-max neural networks—Part 1: Classification. *IEEE Transactions on Neural Networks*, 3(5):776–786, 1992.

[8] S. Abe and M.-S. Lan. A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *IEEE Transactions on Fuzzy Systems*, 3(1):18–28, 1995.

[9] C.-W. V. Chen and T.-Z. Liu. A competitive learning process for hyperspherical classifiers. *Neurocomputing*, 17:99–110, 1997.

[10] P. M. Patil, U. V. Kulkarni, and T. R. Sontakke. Modular fuzzy hypersphere neural network. In *Proceedings of the Twelfth IEEE International Conference on Fuzzy Systems*, volume 1, pages 232–236, 2003.

[11] G. C. Anagnostopoulos and M. Georgiopoulos. Hypersphere ART and ARTMAP for unsupervised and supervised, incremental learning. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, volume 6, pages 59–64, Como, Italy.

[12] S. Abe. *Pattern Classification: Neuro-Fuzzy Methods and Their Comparison*. Springer-Verlag, London, 2001.

[13] S. Abe. Training of support vector machines with mahalanobis kernels. In W. Duch, J. Kacprzyk, E. Oja, and S. Zadrozny, editors, *Artificial Neural Networks: Formal Models and Their Applications (ICANN 2005)—Proceedings of International Conference, Warsaw, Poland*, pages 571–576. Springer-Verlag, Berlin, Germany, 2005.

[14] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, 2001.

[15] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201, 2001.

[16] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7:179–188, 1936.

[17] J. C. Bezdek, J. M. Keller, R. Krishnapuram, L. I. Kuncheva, and N. R. Pal. Will the real iris data please stand up? *IEEE Transactions on Fuzzy Systems*, 7(3):368–369, 1999.

[18] H. Takenaga, S. Abe, M. Takatoo, M. Kayama, T. Kitamura, and Y. Okuyama. Input layer optimization of neural networks by sensitivity analysis and its application to recognition of numerals. *Electrical Engineering in Japan*, 111(4):130–138, 1991.

[19] S. M. Weiss and I. Kapouleas. An empirical comparison of pattern recognition, neural nets, and machine learning classification methods. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 781–787, Detroit, 1989.

[20] 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, 1988.

[21] 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, 1994.