Available online at www.ijpe-online.com

International Journal of Performability Engineering

vol. 15, no. 6, June 2019, pp. 1611-1619
DOI: 10.23940/ijpe.19.06.p12.16111619

Piecewise Combination of Hyper-Sphere Support Vector Machine for Multi-Class Classification Problems

Shuang Liu[a,*], Peng Chen[b], Jiayi Li[a], Hui Yang[a], and Niko Lukač[c]

[a]*School of Computer Science and Engineering, Dalian Minzu University, Dalian, 116600, China*
[b]*Department of Software Engineering, Dalian Neusoft University of Information, Dalian, 116023, China*
[c]*Faculty of Electrical Engineering and Computer Science, University of Maribor, Maribor, SI-2000, Slovenia*

Abstract

Hyper-sphere Support vector machine (SVM) is a widely used machine learning method for multi-class classification problems such as image recognition, text classification, or handwriting recognition. In most cases, only one hyper-sphere optimization problem is computed to solve the problem. However, there are many complex applications with complicated data distributions. In these cases, the computation cost will be increased with unsatisfied classification results if only one support vector machine is adopted as the classification decision rule. To achieve good classification performance, a piecewise combination of the hyper-sphere support vector machine is put forward in this paper based on the analysis of the data sample distribution. First, statistical analysis is adopted for the original data. Then, the *k*-means cluster algorithm is introduced to compute cluster centers for different classes of the data. For the *n* classes classification problem, *m* ($m > n$) hyper-spheres are computed to solve the objective problems based on the number of data centers. For simple sphere-distribution and locally linearly separable distribution cases, the minimum enclosing and maximum excluding support vector machine and the combination of hyper-sphere support vector machine are defined. Experimental results show that different support vector machines for different data distributions will improve the final classification performance.

Keywords: support vector machine; hyper-sphere; data distribution; piecewise; combination; classification performance

(Submitted on March 20, 2019; Revised on April 4, 2019; Accepted on June 8, 2019)

1. Introduction

Binary SVM was first put forward in the machine learning field based on statistical theory and has been successfully applied in many real applications [1-2]. Focused on finding a maximum-margin hyper-plane separating training samples, binary SVM tries to maximize the width of the gap between the two classes. For a new test sample, its classification decision can be decided by this optimal hyper-plane restricted with support vectors. In real applications, it is hard to classify only two classes. There are many methods to extend binary SVM to multiclass classification [3-4]. For example, a multi-class SVM classifier is defined by combining multiple two-class SVMs. The one-against-all method is a typical example that solves multiclass classification by constructing *k* binary classifiers. In this method, each SVM separates one class with all other remaining classes. One-against-one is another method that constructs $k(k-1)/2$ SVM classifiers, with each SVM separating a pair of classes. The shortage for these methods is the high computation cost, because at least $k(k-1)/2$ quadratic programming (QP) optimization problems need to be solved.

Based on one-class SVM [5-7], sphere-structured SVM is presented to solve the problem of high computation costs. The difference between traditional SVM and hyper-sphere SVM is the decision rule. Hyper-sphere SVM substitutes the hyper-plane of binary SVM with a class-specific hyper-sphere. The main idea of a sphere-structured SVM is to find a minimum bounding hyper-sphere, which restricts all training samples belonging to one class within the hyper-sphere as much as possible. Restrictions for each bounding hyper-sphere are its center and its radius. Similar with the idea of maximum-margin in binary SVM, this method adopts the smallest radius to get the maximized gap between different bounding hyper-spheres.

* Corresponding author.
E-mail address: liushuang@dlnu.edu.cn

Because the resulting hyper-spheres for each class is obtained based on direct computation of quadratic programming (QP) optimization, its computation cost is relatively lower than that of the other combination methods.

In most cases, only one objective function is adopted, not considering data distribution rules. However, data distribution rules are very complicated in many applications. One objective function may lead to almost all training samples being computed as support vectors, increasing the computation cost for the objective function. Thus, the computation cost for a new test sample is high, with unsatisfied classification accuracy due to solution solving with all support vectors. To solve this problem, a piecewise combination of hyper-sphere SVM is put forward, taking all data distribution rules into consideration and adopting different decision rules to improve the classification performance. The main idea of our method is to get statistical rules for the original training dataset, compute clustering centers, and decide the objective function based on sample distribution to get the satisfied classification accuracy in real applications.

The remainder of the present study is structured as follows. Section 2 gives a summarization of SVM and sphere-structured SVM. Section 3 introduces the statistical analysis of data distribution, mathematical description of our piecewise combination of hyper-sphere SVM, and framework of the proposed method. To verify the proposed method, Section 4 discusses some experimental results. Section 5 gives the conclusions.

2. Related Work

The classification problem to solve real applications can be divided into four steps: data preprocessing, feature extraction and feature selection, objective function solving, and classification accuracy test to improve the objective function. The first two steps are not our focus, and our study started with a normalized training dataset and testing dataset.

Many studies have focused on extending binary SVM to multi-class classification problems. Vural and Dy [8] introduced a framework called divide-by-2 (DB2) to extend support vector machines (SVM) to multi-class problems. Mele and Maver [9] used hierarchical SVMs to complete object recognition. Benabdeslem et al. [10] proposed a dendogram-based SVM for multi-class classification. Chmielnicki et al. [11] combined 1-v-1 and 1-v-r strategies to improve the multiclass SVM classifier. Le et al. [12] proposed a theoretical framework for multi-sphere support vector data description. Cevikalp and Triggs [13] adopted cascades of binary and one-class classifiers to complete visual object detection.

Sphere-structured SVM has been studied in-depth since its introduction. Xiao et al. [14] studied how to detect outliers on multi-distribution data with multi-sphere support vector data description. Liu et al. [15] discussed minimum enclosing and maximum excluding machine for pattern description and discrimination. Yang et al. [16] adopted an unsupervised quarter-sphere support vector machine to complete the online outlier detection technique for wireless sensor networks. Liu et al. [17-18] proposed a fuzzy hyper-sphere SVM and a multiple sub-hyper-spheres SVM for multi-class classification.

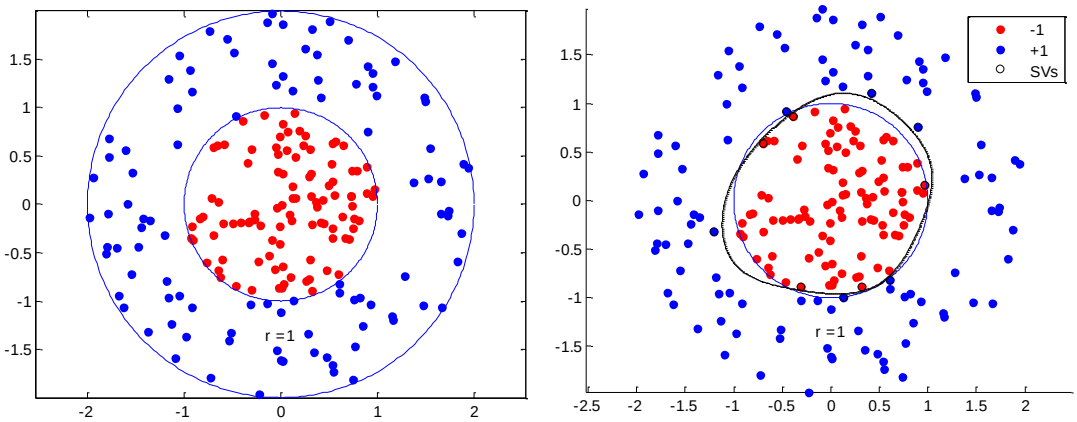
Multi-class classification SVM has been widely applied in the fields of text categorization, sounds recognition, image processing, and image annotation and has achieved satisfying results. Rabaoui [19] presented an improved one-class SVM classifier for sound classification. Jordi et al. [20] applied semisupervised one-class support vector machines for classification of remote sensing data. Mohanty [21] utilized SVM to identify speakers during Oriya speech recognition. Sun et al. [22] provided a comparative study of imbalanced text classification using SVM.

3. Method

3.1. Analysis of Data Distribution

There are three cases for data distribution for samples: sphere-inclusion-distribution, multi locally centered distribution, and common distribution. In many cases, only one SVM objective function is adopted to solve the classification problem, not considering data distribution rules, which will lead to an increase in computation cost with all training samples as support vectors. Classification accuracy for new test points may be unsatisfied because of the training process.

(1) Samples satisfy the sphere-inclusion-distribution if two data centers are very close, the data in each class is evenly distributed, and data of one class is included in another class. The simplest example is shown in Figure 1. With hyper-sphere SVM as the objective function, two bounding hyper-sphere are obtained to decide the classification for samples. For this binary classification example, the solution of traditional binary SVM is a nearly circular curve with many support vectors, as shown in Figure 1(b). The difference between hyper-sphere SVM and traditional binary SVM is the number of support vectors, which will increase the computation cost for new test points.



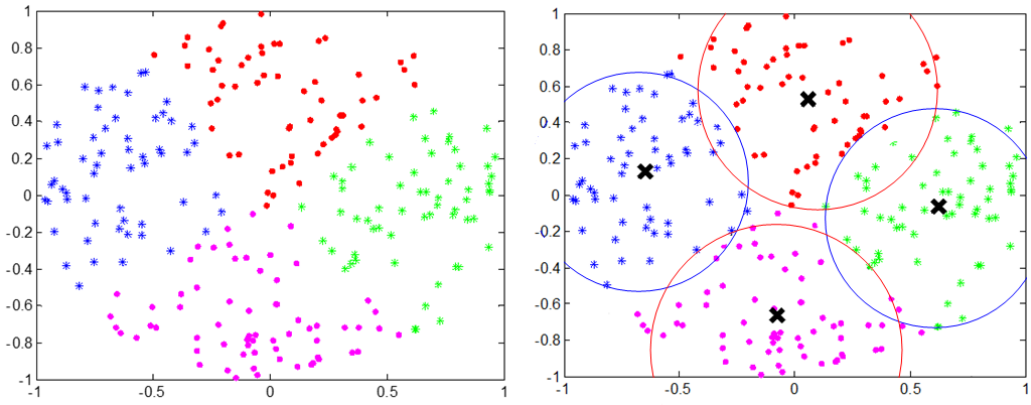
(a) Hyper-sphere SVM result for 2D samples (b) Traditional binary SVM result for the same samples

Figure 1. An example of sphere-inclusion distribution in 2D case

For sphere-inclusion-distribution case, points coordinates are sorted first based on mathematical statistical tools. Then the two-means clustering method is adopted to get two data centers, comparing their distance difference. If data points for each class in each dimension are evenly distributed by counting data points in each dimension and sorted by some rules, the data of the two classes is considered a sphere-inclusion distribution.

(2) Samples satisfy the multi locally centered distribution if there are multi locally centers for one class data and there are no other fixed distribution rules for the data of this class. The solution process begins with sorting points' coordinates. Then, the *k*-means clustering method is adopted to get clustering centers. If there is more than one center for samples of one class, multiple hyper-sphere SVM objective functions are computed to get the final resulting decision spheres.

Figure 2 shows a simple example in 2D. Blue asterisk marks and green asterisk marks belong to class 1. Red dots and pink dots belong to class 2. The original dataset is shown in Figure 2(a). There are four data centers marked with a black '×'. Four bounding spheres (circles) are computed to obtain the final classification result, as shown in Figure 2(b). Two blue bounding spheres are for class 1, and two red bounding spheres are for class 2. For a new test point, it is easy to decide its classification based on the four spheres with good classification performance.



(a) Original data for two classes (b) Classification result for two classes with four centers

Figure 2. An example of four hyper-spheres for two classes with four data centers in 2D

(3) Samples are considered common distribution if no distribution rules can be found based on mathematical statistical analysis. Only common hyper-sphere SVM is adopted as the objective function. The special case for this decision process is that the spheres are intersected with each other. Data distribution in the intersections are analyzed first, and then the decision class is decided by different strategies. If the training samples of two classes in the intersection can be classified by intersecting the hyper-plane with two hyper-spheres, then new test samples can be decided by this plane. If the training samples of two classes in the intersection can be approximately linearly classified, then new test samples can be classified by the standard optimal binary-SVM hyper-plane. If the training samples of two classes in the intersection cannot be linearly classified, then new test samples can be decided by introducing a kernel function to obtain the optimal classification hyper-plane. If the training examples belong to only one class, then new test samples can be classified by the exclusion method.

3.2. Mathematical Description of Piecewise Combination of Hyper-Sphere Support Vector Machine

Without loss of generalization, slack variables ξ_i are introduced by permitting isolated points, and a nonlinear mapping ϕ is introduced by transforming the training samples into a high dimensional feature space to solve nonlinear separation problems and then compute the hyper-sphere in the feature space. Suppose there are *n* dimensional training samples of *m* classes to be classified. The objective is to solve the *k*-class classification problem, that is, $X_k = \{\boldsymbol{x}_i \in R^n, i = 1, \cdots, l_k\}$, $(k = 1, \cdots, m_{k1}, m_{k2}, \cdots, m_{ks})$. For a class *m*, there may be multiple corresponding minimum bounding hyper-spheres, such as S_{k1}, S_{k2}, and S_{ks}. Then, finding the minimum bounding hyper-sphere of each class that encloses all the training examples of that class can be computed by solving the following constrained quadratic optimization problem in Equation (1).

$$\begin{aligned} &\min_{c_{kt}, R_{kt}} \ R_{kt}^2 + C_{kt} \sum_{i=1}^{l_{kt}} \xi_i \\ &s.t. \\ &\left\| \phi(\boldsymbol{x}_i) - c_{kt} \right\|^2 \le R_{kt}^2 + \xi_i \\ &\xi_i \ge 0, \ \ i = 1, \cdots, l_{kt} \\ &t = 1, \cdots, s \end{aligned} \tag{1}$$

For class *k*, the minimum bounding hyper-sphere S_{kt} is decided by its center c_{kt} and radius R_{kt}. *s* is the number of cluster centers for class *k*. C_{kt} is the penalty factor, and $\xi_i \ge 0$ are slack variables. The difference between common hyper-sphere SVM and our piecewise combination of hyper-sphere SVM is the number of minimum bounding spheres for one class. By introducing Lagrange multipliers α_i, β_i, the Lagrange polynomial can be written as Equation (2).

$$L(R_{kt}, c_{kt}, \xi_i, \alpha_i, \beta_i) = \ R_{kt}^2 + C_{kt} \sum_{i=1}^{l_{kt}} \xi_i - \sum_{i=1}^{l_{kt}} \alpha_i (R_{kt}^2 + \xi_i - \left\| \phi(\boldsymbol{x}_i) - c_{kt} \right\|^2) - \sum_{i=1}^{l_{kt}} \beta_i \xi_i \tag{2}$$

Equations (3) to (5) can be obtained by taking the partial directives of *L* with respect to R_{kt}, c_{kt}, ξ_i.

$$\frac{\partial L}{\partial R_{kt}} = 2R_{kt} - 2R_{kt} \sum_{i=1}^{l_{kt}} \alpha_i = 0 \Rightarrow \sum_{i=1}^{l_{kt}} \alpha_i = 1 \tag{3}$$

$$\frac{\partial L}{\partial c_{kt}} = 2 \sum_{i=1}^{l_{kt}} \alpha_i \phi(\boldsymbol{x}_i) - 2c_{kt} \sum_{i=1}^{l_{kt}} \alpha_i = 0 \Rightarrow c_{kt} = \frac{\sum_{i=1}^{l_{kt}} \alpha_i \phi(\boldsymbol{x}_i)}{\sum_{i=1}^{l_{kt}} \alpha_i = 1} = \sum_{i=1}^{l_{kt}} \alpha_i \phi(\boldsymbol{x}_i) \tag{4}$$

$$\frac{\partial L}{\partial \xi_i} = C_{kt} - \alpha_i - \beta_i = 0 \Rightarrow \alpha_i = C_{kt} - \beta_i \Rightarrow 0 \le \alpha_i \le C_{kt} \tag{5}$$

Substituting them back into Equation (2), the original optimization problem becomes its dual optimization problem, with the format shown in Equation (6).

$$\begin{aligned} &\min_{\alpha_i} \ \sum_{i,j=1}^{l_{kt}} \alpha_i \alpha_j \mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) - \sum_{i=1}^{l_{kt}} \alpha_i \mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}_i) \\ &s.t. \ \ \sum_{i=1}^{l_{kt}} \alpha_i = 1 \\ &\qquad 0 \le \alpha_i \le C, \ i = 1, \cdots, l_{kt} \end{aligned} \tag{6}$$

Without computing the introducing nonlinear map function, the kernel trick is adopted by computing inner products in the feature space, that is, $\mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$. The Lagrange multipliers can be obtained by solving the dual quadratic programming problem. Support vectors are the vectors $\boldsymbol{x}_i$ with $\alpha_i > 0$. Based on the above deduction process, the center c_{kt} can be computed by $c_{kt}^2 = \sum_{i=1,j}^{l_k} \alpha_i \alpha_j \mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The resulting decision function can be computed as Equation (7).

$$f_{kt}(\boldsymbol{x}) = \mathrm{sgn}(R_{kt}{}^2 - \sum_{i,j=1}^{l_{kt}} \alpha_i \alpha_j \mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}_j) + 2\sum_{i=1}^{l_{kt}} \alpha_i \mathrm{K}(\boldsymbol{x}_i, \boldsymbol{x}) - \mathrm{K}(\boldsymbol{x}, \boldsymbol{x})) \tag{7}$$

For any support vector, the radius R_{kt} can be computed by equating Equation (7) to zero. If $f_{kt}(\boldsymbol{x}) > 0$, the new point $\boldsymbol{x}$ falls inside of the hyper-sphere based on Equation (7). Similarly, $\boldsymbol{x}$ falls outside of the hyper-sphere if $f_{kt}(\boldsymbol{x}) < 0$ and $\boldsymbol{x}$ lies on the hyper-sphere if $f_{kt}(\boldsymbol{x}) = 0$. In 2D cases, the bounding sphere is a circle. In 3D cases, the bounding spheres are like bubbles, as shown in Figure 3. There are five spheres, corresponding to five classes.

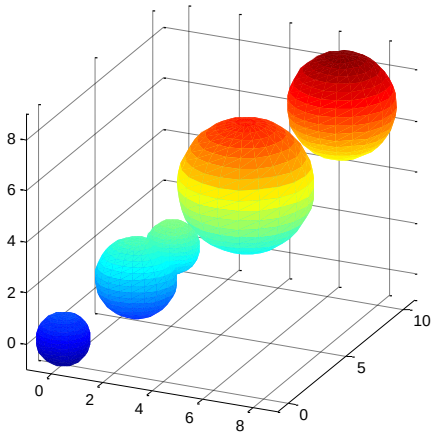


Figure 3. Hyper-sphere SVM result for 3D samples

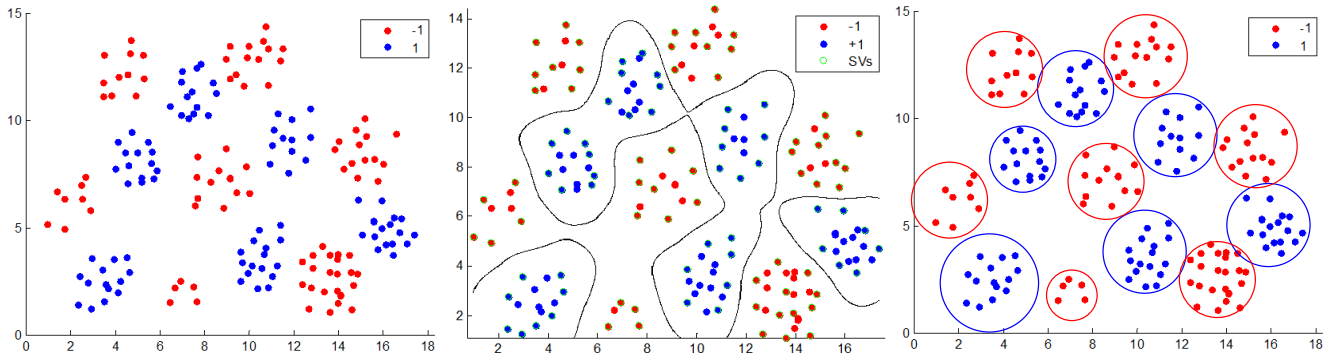
3.3. Framework of the Proposed Method

The framework of the proposed method consists of three processes: data preprocessing, hyper-sphere solving, and classification testing. It is often applied to obtain classifications in the field of multiple classification. The proposed method follows eight steps to complete the classification process.

- **Step 1** All training samples are sorted by their coordinates. Feature extraction and selection are not our consideration.
- **Step 2** *k* cluster centers are computed by the *k*-means clustering algorithm.
- **Step 3** Binary SVM is adopted to get the optimal hyper-plane if *k* = 2. The algorithm stops if the classification accuracy for test samples are acceptable; otherwise, it proceeds to Step 4. If *k* > 2, it proceeds to Step 5.
- **Step 4** *k* = 2, and the distance between two clusters is lower than the predefined threshold. If the samples of two classes in each dimension are evenly distributed by counting samples in each dimension, the data of the two classes is considered a sphere-inclusion distribution. Simple hyper-sphere SVM is called to the optimal hyper-sphere. If the classification accuracy for the test samples meets the precision requirement, the algorithm proceeds to Step 7; otherwise, it returns to Step 3 after re-computation of two subsets of the original training samples.
- **Step 5** Class labels are marked for *k* cluster centers. For example, five centers are denoted as $y_{11}, y_{12}, \cdots, y_{15}$ if there are five cluster centers for class 1. Similarly, five centers for class 2 are denoted as $y_{21}, y_{22}, \cdots, y_{25}$.
- **Step 6** Each cluster center is compared with its neighbor cluster centers. Centers of the same class are computed together to get the combination of hyper-sphere SVMs.
- **Step 7** For new samples or test samples, Euclidean distances between them with *k* clustering centers are computed to choose the combination of hyper-sphere SVMs. The classification result is the maximization of the decision function.
- **Step 8** If classification accuracy for test samples meets the precision requirement, the algorithm stops. Otherwise, the subsets of the two closest clustering centers are chosen to continue the computation process, and the algorithm returns to Step 4.

4. Experiments

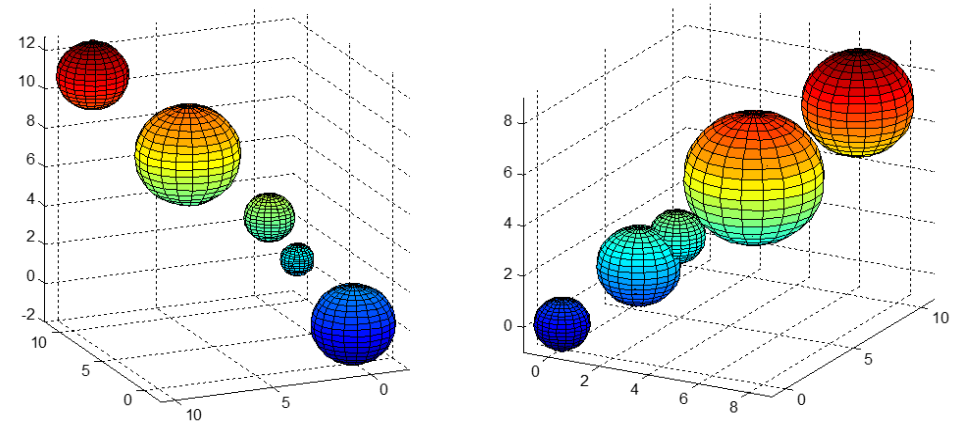
To show the classification result of our method, a simulation example is given as follows. There are only data coming from two classes in Figure 4. Red points belong to one class, and blue points belong to the other class. Clearly, these data points are nonlinear and separable. The classification result based on traditional SVM with polynomial kernel is shown in Figure 4(b). If a 2D hyper-sphere SVM is adopted, the result shown in Figure 4(c) is achieved. Its computation cost is lower than traditional binary SVM. As shown in Figure 4(b), almost every point are considered as support vectors. So the decision for future test point consists of computation for all points. For a new test point for the bounding sphere, only one sphere is needed to get the final classification result. So the classification performance is better for future classification than polynomial result.



(a) Data distribution for a simulation example in 2D (b) Traditional SVM classification result (c) Multiple hyper-spheres for the case

Figure 4. The original cell image and the segmented cell image

Figure 5 shows a 3D example. In 3D cases, the bounding circles change into spheres. If all the data coming from five classes are independent, their spheres are also independent, as shown in Figure 5(a). If the data coming from two classes are intersected, the resulting hyper-spheres are also intersected, as shown by the two middle blue spheres in Figure 5(b). For a new test point in case of Figure 5(b), some decision strategy is adopted to decide its classification result, as in [17].



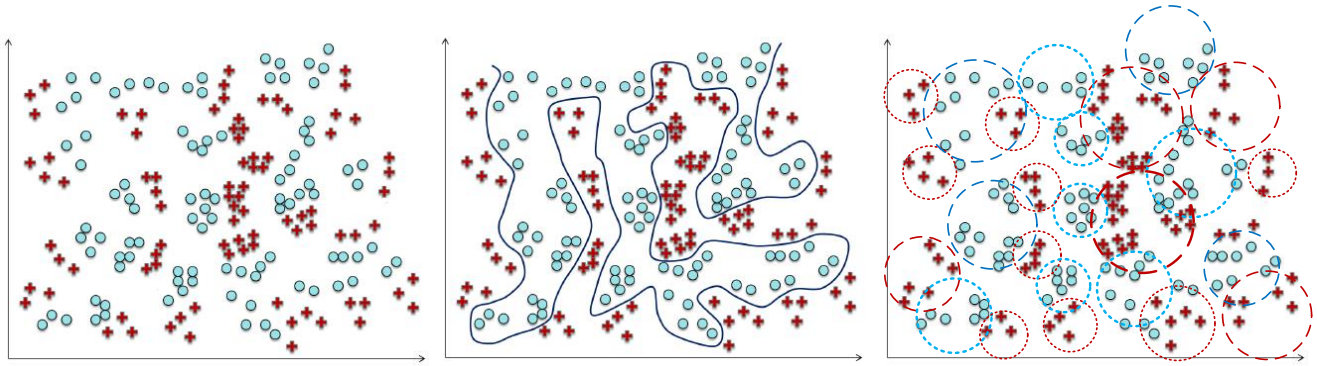
(a) Five independent hyper-spheres for five classes (b) Two intersected hyper-spheres for five classes

Figure 5. An example of five classes classification in 3D case

Figure 6 shows a real application example. It is a gene classification example after feature extraction and normalization. As shown in Figure 6(a), these points are relatively complicated because there are no obvious distribution rules. A classification result based on traditional SVM with RBF Gaussian kernel is shown in Figure 6(b). This curve is complicated as well, with many points as support vectors. As shown in Figure 6(c), many bounding spheres are obtained if the threshold for data centers is relatively small. However, for a new test point, its decision is only dependent on five or ten points. The computation cost is relatively lower than that of the curve in Figure 6(b).

Another real application example is an action recognition system from videos. The training dataset comes from camera data placed in our university and some special recorded videos by our students. With the data preprocessing of foreground extraction and morphological operations optimization, features such as trajectory features, shape features, and KLT tracking

points are extracted for each action recognition. These features are input into the proposed piecewise combination of hyper-sphere SVMs. With these training samples, normal actions can be recognized by our hyper-sphere SVM classifier, and new abnormal actions can be recognized in future videos. Because most of the actions captured in videos belong to normal actions, it is easy to recognize normal actions. In our dataset, normal actions consist of walking, running, swimming, standing, and so on. Recognized actions of fighting and running are shown in Figure 7.



(a) Data distribution for a real application in 2D case (b) Traditional SVM classification result (c) Multiple hyper-spheres for the case

Figure 6. A real application example based on traditional SVM and our multiple hyper-sphere SVM

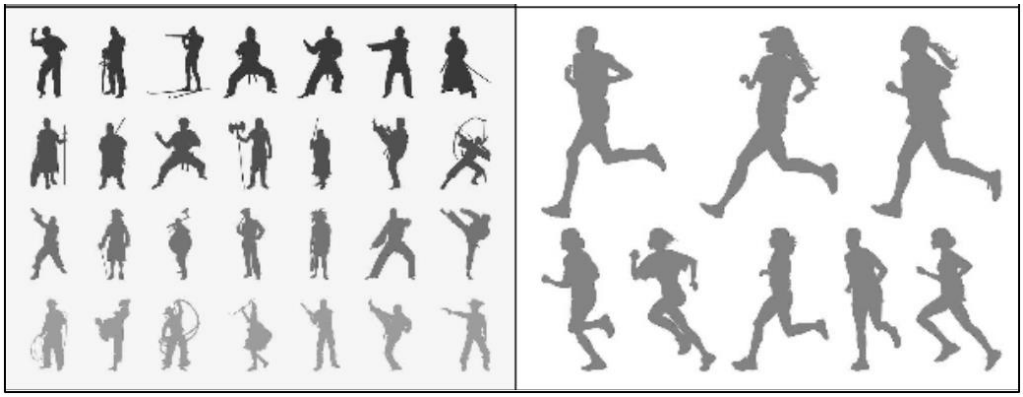


Figure 7. A real application example of action recognition for videos

Table 1 shows the comparison results between our method and the traditional combination of binary SVMs. The number of training samples is 2500, and testing samples is 500. Training data and testing data were randomly selected for each experiment. All experiments were performed ten times, and the data in Table 1 shows the average value of each indicator.

Table 1. Comparison of action recognition results

Type of Algorithm	Training Time	No. of Support Vector	Accuracy (%)
One-against-one binary SVM	1997s	1145	78.6
One-against-all binary SVM	1982s	1235	76.3
Combination of binary SVMs	1927s	1032	82.1
Common hyper-sphere SVM	1752s	692	85.6
Our piecewise combination of hyper-sphere SVMs	1306s	580	89.6

5. Conclusions

Compared with combining binary SVM classifiers to solve multi-class classification problems, sphere-structured SVM is not only convenient and saves computation costs but also reduces the error probability for new test samples. The present study focuses on the data distribution rules and corresponding optimized objective function. Training samples follow three main data distribution rules. A piecewise combination of hyper-sphere SVMs is proposed to improve the classification performance. The experimental results demonstrate the effectiveness of the proposed method compared with the results based on combination of binary SVMs. The effectiveness of different types of support vector machines is also analyzed based on these results, and final classification performance is improved.

Acknowledgments

This work is partially supported by the National Nature Science Foundation of Liaoning Province (No. 2015020099) and the National Natural Science Foundation (No. 71303031).

References

1. V. N. Vapnik, "An Overview of Statistical Learning Theory," *IEEE Transactions on Neural Networks*, Vol. 10, No. 5, pp. 988-999, 1999
2. V. N. Vapnik, "The Nature of Statistical Learning Theory," Springer-Verlag, New York, 1999
3. J. Weston and C. Watkins, "Support Vector Machines for Multi-Class Pattern Recognition," in *Proceedings of the European Symposium on Artificial Neural Networks*, pp. 219-224, Bruges, 1999
4. C. W. Hsu and C. J. Lin, "A Comparison of Methods for Multi-Class Support Vector Machines," *IEEE Transactions on Neural Networks*, Vol. 13, No. 2, pp. 415-425, 2002
5. M. L. Zhu, S. F. Chen, and X. D. Liu, "Sphere-Structured Support Vector Machines for Multi-Class Pattern Recognition," *Lecture Notes in Computing Science*, Vol. 2369, pp. 589-593, 2003
6. Q. Wu, C. Y. Jia, and A. F. Zhang, "An Improved Algorithm based on Sphere Structure SVMs and Simulation," *Journal of System Simulation* (*Chinese*), Vol. 20, No. 2, pp. 345-348, 2008
7. Y. Chen, X. S. Zhou, and T. S. Huang, "One-Class SVM for Learning in Image Retrieval," in *Proceedings of the 2001 IEEE International Conference On Image Processing*, pp. 34-37, 2001
8. V. Vural and J. G. Dy, "A Hierarchical Method for Multi-Class Support Vector Machines," in *Proceedings of International Conference on Machine Learning*, pp. 831-838, ACM, 2004
9. K. Mele and J. Maver, "Object Recognition using Hierarchical SVMs," in *Proceedings of Computer Vision Winter Workshop '03*, Valtice, Czech Republic, 2003
10. K. Benabdeslem and Y. Bennani, "Dendogram-based SVM for Multi-Class Classification," *Journal of Computing & Information Technology*, Vol. 14, No. 4, pp. 283-289, 2006
11. W. Chmielnicki and K. Stąpor, "Combining One-Versus-One and One-Versus-All Strategies to Improve Multiclass SVM Classifier," in *Proceedings of the 9th International Conference on Computer Recognition Systems*, pp. 37-45, 2016
12. T. Le, D. Tran, W. Ma, and D. Sharma, "A Theoretical Framework for Multi-Sphere Support Vector Data Description," in *Proceedings of the 17th International Conference on N Neural Information Processing Models & Applications ICONIP'10*, Vol. part II, pp. 132-142 , Sydney, Australia, 2010
13. H. Cevikalp and B. Triggs, "Visual Object Detection using Cascades of Binary and One-Class Classifiers," *International Journal of Computer Vision*, Vol. 123, No. 3, pp. 334-349, 2017
14. Y. S. Xiao, B. Liu, L. B. Cao, X. D. Wu, C. Q. Zhang, Z. F. Hao, et al., "Multi-Sphere Support Vector Data Description for Outliers Detection on Multi-Distribution Data," in *Proceedings of 2009 IEEE International Conference on Data Mining Workshops*, pp. 82-87, 2009
15. Y. Liu and Y. F. Zheng, "Minimum Enclosing and Maximum Excluding Machine for Pattern Description and Discrimination," in *Proceedings of the 18th International Conference on Pattern Recognition*, Vol. 3, pp. 129-132, 2006
16. Z. Yang, N. Meratnia, and P. Havinga, "An Online Outlier Detection Technique for Wireless Sensor Networks using Unsupervised Quarter-Sphere Support Vector Machine," in *Proceedings of International Conference on Intelligent Sensors*, pp. 151-156, 2008
17. S. Liu, P. Chen, and K. Q. Li, "Multiple Sub-Hyper-Spheres Support Vector Machine for Multi-Class Classification," *International Journal of Wavelets Multiresolution and Information Processing*, Vol. 12, No. 3, 2014
18. S. Liu, P. Chen, and J. Yun, "Fuzzy Hyper-Sphere Support Vector Machine for Pattern Recognition," *ICIC Express Letters*, Vol. 9, No. 1, pp. 87-92, 2015
19. A. Rabaoui, M. Davy, S. Rossignol, Z. Lachiri, and N. Ellouze, "Improved One-Class SVM Classifier for Sounds Classification," in *Proceedings of IEEE Conference on Advanced Video & Signal based Surveillance*, pp. 117-122, 2007
20. J. Munoz-Mari, F. Bovolo, L. Gomez-Chova, L. Bruzzone, and G. Camp-Valls, "Semisupervised One-Class Support Vector Machines for Classification of Remote Sensing Data," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 48, No. 8, pp. 3188-3197, 2010
21. S. Mohanty, "Speaker Identification using SVM During Oriya Speech Recognition," *I. J. Image, Graphics and Signal Processing*, Vol. 10, pp. 28-36, 2015
22. A. Sun, E. P. Lim, and Y. Liu, "On Strategies for Imbalanced Text Classification using SVM: A Comparative Study," *Decision Support Systems*, Vol. 48, No. 1, pp. 191-201, 2009

Shuang Liu received her Ph.D. in traffic information engineering and control from Dalian Maritime University, China. She is currently an associate professor at Dalian Minzu University, China. Her current research interests include machine learning and image processing.

Peng Chen is currently a professor at Dalian Neusoft University of Information, China. His research interests include machine learning, collision avoidance, and intelligent information processing.

Jiayi Li is currently a postgraduate candidate at Dalian Minzu University, China. Her research interests include deep learning and big data.

Hui Yang is currently a postgraduate candidate at Dalian Minzu University, China. His research interests include deep learning and big data.

Niko Lukač obtained his Ph.D. in computer science in 2016 from Maribor University. He is currently a researcher in the Faculty of Electrical Engineering and Computer Science at the University of Maribor, Slovenia.