Feature Extraction based on Space Folding Model and Application to Machine Learning

Minh Tuan Pham[1], Kanta Tachibana[2], Tomohiro Yoshikawa[1] and Takeshi Furuhashi[1]
[1]Nagoya University, Furo-cho, Chikusa-ku, Nagoya 464-8603, Japan
[2]Kogakuin University, 1-24-2 Nishi-Shinjuku, Tokyo, 163-8677, Japan

***Abstract*—One of the most important designs for a lot of machine learning methods is the determination of the similarity between instances. Especially the kernel matrix, which is also known as the Gram matrix, plays a central role in the kernel machines such as support vector machine. The simplest design of similarity function is to use the distances between instances or the Gaussian function based on them. It is easy to learn the model when the data distribution follows their label, in which the instances with same label are allocated near and those with different label are allocated far. However, when the data distribution is non-linear, it becomes difficult. This paper discusses the inner products of 2 non-orthogonal basis vectors and proposes the similarity between instances. This paper also proposes a space folding model for machine learning based on the proposed similarity. This paper applies the proposed method to pattern recognition problem and shows its effectiveness.**

I. Introduction

For more efficient machine learning, the feature extraction is important for a lot of machine learning methods such as Support Vector Machine(SVM)[1] or Neural Networks(NN)[2]. When the data distribution itself is not linearly separable some kinds of non-linear transformation is used to make the feature space as separable as possible. To do this, a lot of conventional machine learning methods considered similarity between instances. Especially the kernel matrix, which is also known as the Gram matrix[3], plays a central role in the kernel machines such as SVM. The simplest design of similarity function is to use the Euclidean distances between instances or the Gaussian function based on them. It is easy to learning the model if the data distribution follows that the instances with the same label are allocated near and those with the different label are allocated far. However, when the data distribution of same class are divided into plural clusters, it becomes difficult for learning.

For more efficient machine learning, the conventional methods extracted the feature from data before designing of the model. It is a popular approach to use linear methods such as principal component analysis (PCA)[4] or multiple discriminant analysis (MDA)[5] to extract the feature from the data distribution. PCA calculates an eigen-value decomposition of a data covariance matrix. MDA finds a linear combination of features which best separates two or more classes of objects. But, because these linear methods do not consider the relations between the distances and the labels of the instances, they are not appropriate for designing the model when the data distribution is not linear. In other hands, the conventional methods use nonlinear feature extraction based on kernel feature space such as kernel principal component analysis (KPCA)[6] or kernel orthonormalized partial least squares (KOPLS)[7]. These methods used the Gaussian function for designing the kernel. But it is difficult to optimize the parameter of the Gaussian function and it cannot change the topology of data distribution.

This paper proposes a space folding model which can change the similarity (distance) between the instances, and applies it to the classification problems. The proposed method divides each basis vector into positive and negative directions and optimize $2m$ vectors (Space Folding Vectors: SFV) in the m dimensional space. Then, the proposed method estimates the $2m$ SFVs so that distance between instances with the same label becomes smaller, and distance between those with different labels becomes larger by minimizing the cross entropy[8][9] between the labels and the distances. Because the proposed method linearly transforms each quadrant differently from other quadrants and it can change the topology of data distribution, it is expected to improve classification performance in comparison with the feature extraction by the conventional linear or nonlinear transformations with kernel methods.

This paper shows the effectiveness of the proposed method through two experiments of classification. One is those prepared as a preliminary experiment. The experiment results show that the proposed space folding model is effective for machine learning in a case of symmetric data. And in a case of non-symmetric data, this experiment also shows the proposed model using a folding point at the center of the class with the largest variance is effective too. The other experiment is to classify the hand-written digits dataset of the UCI Machine Learning Repository [10]. Then this experiment shows that the classification rate using the proposed space folding model is better than without using it.

II. Proposed Method

A. Inner Product of Basis Vectors and Distance between Instances

This section explains the definition of inner product of 2 basis vectors. Using an orthonormal basis $\{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_m\}$ which be chosen for a real vector space $\mathbf{R}^m$, the inner product

of $\mathbf{e}_i$ and $\mathbf{e}_j$ is defined by

$$\mathbf{e}_i \cdot \mathbf{e}_j = \begin{cases} 1 & (i = j)\,, \\ 0 & (i \neq j)\,. \end{cases} \tag{1}$$

The distance between two instances $k, l \in \{1, \ldots, n\}$ is defined as

$$\begin{aligned} d_{k,l}^2 &= \|\mathbf{x}_k - \mathbf{x}_l\|^2, \\ &= \|\sum_i^m (x_{k;i} - x_{l;i})\,\mathbf{e}_i\|^2, \\ &= \sum_i^m \sum_j^m (x_{k;i} - x_{l;i})\,(x_{k;j} - x_{l;j})\,\mathbf{e}_i \cdot \mathbf{e}_j, \end{aligned} \tag{2}$$

where, $\mathbf{x}_k = x_{k;1}\mathbf{e}_1 + \cdots x_{k;d}\mathbf{e}_m = \sum_i^m x_{k;i}\mathbf{e}_i$ is the coordinate of the instance k in m dimensional space. From definition (1), the distance is calculated as $d_{k,l}^2 = \sum_i^m (x_{k;i} - x_{l;i})^2$, which is also used in conventional methods. This paper utilizes non-orthogonal basis vectors. Their inner product becomes

$$\mathbf{e}_i \cdot \mathbf{e}_j = \kappa_{i,j} \in \mathbb{R}. \tag{3}$$

Using these non-orthogonal vectors, the distance of 2 instances is calculated as

$$\begin{aligned} d_{k,l}^2 &= \sum_i^m \sum_j^d (x_{k;i} - x_{l;i})\,(x_{k;j} - x_{l;j})\,\mathbf{e}_i \cdot \mathbf{e}_j \\ &= \sum_i^d \sum_j^m (x_{k;i} - x_{l;i})\,(x_{k;j} - x_{l;j})\,\kappa_{i,j} \\ &\neq \sum_i^m (x_{k;i} - x_{l;i})^2\,, \end{aligned} \tag{4}$$

which shows that the distance of 2 instances depends on $\kappa_{i,j}$.

B. Space Folding Model

This section describes space folding model for classification problem. The problem is to infer labels for unknown unlabeled data when the labels $\{y_k | k \in N\}$ of an instance set $N = \{1, \ldots, n\}$ are known. It is easy to be solved if the instances with the same label are allocated near and those with the different label are allocated far. However, when the data is not linear, the problem becomes more difficult. For example, the linear method such as Linear Discriminant Analysis (LDA) can not classify correctly when the data distribution of same class are divided into plural clusters. It is able to easy classify them to two classes if the data with same label gather and the distance between the same label data becomes shorter than distances from the different label data.

This paper describes how to define the distance between two instances $k, l \in N$ based on the proposed space folding model using the non-orthogonal vectors. Fig. 1 shows the image of space folding model. The top shows the data distribution in the original space. And the bottom shows the data distribution after folding the space. The proposed method divides each basis vector in the m dimensional space into positive and negative directions and optimize the SFVs to gather the same label data each other.

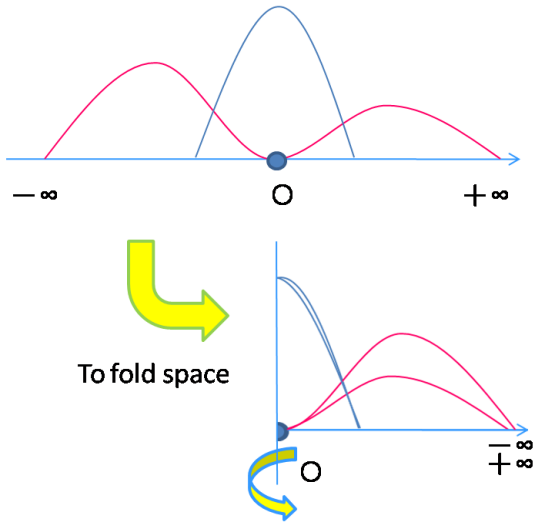

Fig. 1. Image of space folding model.

For a given set of vectors $\{\mathbf{x}_k = \sum_i^m x_{k;i}\mathbf{e}_i \in \mathbf{R}^m, k = 1, \ldots, n\}$ in m-dimensional space, the proposed method divide axes into negative parts and positive parts. It makes the $2m$ vectors (SFVs) in m-dimension. Then, the vector k of SFVs be defined as,

$$\chi_k = \sum_i^m |x_{k;i}| \mathbf{e}_i^* \in \mathbf{R}^m. \tag{5}$$

$$\mathbf{e}_i^* = \begin{cases} \mathbf{e}_i & (x_{k;i} \geq 0)\,, \\ \mathbf{e}_{i+m} & (x_{k;i} < 0)\,. \end{cases} \tag{6}$$

Where, $\mathbf{e}_i \in \mathbb{R}^m, i \in \{1, \ldots, 2m\}$ is an SFV which is defined on m-dimensional space. Then this paper rewrite the vector χ_k as a linear combination of $\mathbf{e}_i, i \in \{1, \ldots, 2m\}$, with weight given by,

$$\xi_{k;i} = \begin{cases} |x_{k;i}| & (x_i\,(2m + 1 - 2i) \geq 0)\,, \\ 0 & (x_i\,(2m + 1 - 2i) < 0)\,. \end{cases} \tag{7}$$

Therefore,

$$\chi_k = \sum_i^{2m} \xi_{k;i}\mathbf{e}_i \in \mathbf{R}^m. \tag{8}$$

The problem is how to find the SFV $\mathbf{e}_i, i \in \{1, \ldots, 2m\}$. This paper estimate $\mathbf{e}_i$ by minimizing the energy

$$E = \sum_{k=1}^{n-1} \sum_{l=k+1}^{n} E_{k,l}, \tag{9}$$

where $E_{k,l}$ is a cross entropy function between two similarity functions $\ell_{k,l}$ and $\rho\,(d_{k,l})$, which is introduced as,

$$E_{k,l} = -\ell_{k,l} \ln \rho\,(d_{k,l}) - (1 - \ell_{k,l}) \ln\,(1 - \rho\,(d_{k,l}))\,. \tag{10}$$

This paper employs $\rho\,(d_{k,l})$ as the similarity function,

$$\rho\,(d_{k,l}) = \exp\left(-\frac{d_{k,l}^2}{2}\right). \tag{11}$$

And $\ell_{k,l}$ is defined as

$$\ell_{k,l} = \begin{cases} 1 & (y_k = y_l), \\ 0 & (y_k \neq y_l). \end{cases} \quad (12)$$

$E_{k,l}$ is approached to minimum where $\ell_{k,l} = \rho(d_{k,l})$. This minimization means the data having the same label are located at the same area. Note that, this minimization uses all data, not only vectors near the boundary but also ones far from the boundary, and it is different from the SVM which uses only support vectors. Therefore, our SFM may not improve the performance of the SVM to the best. But, because our method can gather the same label data each other by optimizing the SFVs, the output of the machine learning including SVM should become better.

C. Algorithm of estimating SFVs

The SFVs of proposed space folding model can be calculated by minimizing the energy function showed as Eq. 9. This paper employs the Newton-Raphson method to estimate $\mathbf{e}_i$. The algorithm of proposed method is as follows:

Step1 Initialize SFVs $\mathbf{e}_1, \ldots, \mathbf{e}_{2m}$.
Step2 Calculate gradient vectors $E_{\mathbf{e}_1}, \ldots, E_{\mathbf{e}_{2m}}$.
Step3 Select $\mathbf{e}_i$ such that $i = \arg\max_j \{\|E_{\mathbf{e}_j}\|^2\}$.
Step4 For the selected SFV, calculate modification vector $\Delta\mathbf{e}_i$, and update $\mathbf{e}_i$ by $\mathbf{e}_i = \mathbf{e}_i + \Delta\mathbf{e}_i$.
Step5 Return to Step2.

In the following, this paper gives more details about each step of the above algorithm.

In the Step1, the simplest initialization way is set $\mathbf{e}_1, \ldots, \mathbf{e}_{2m}$ by random, but it is not an effective idea for learning. This paper initialize the SFVs by folding the orthogonal basis vectors $\{e_j\}$ in original space.

In the case of $i \in \{1, \ldots, m\}$,

$$\mathbf{e}_i = \sum_j^m a_{i;j} e_j \quad (13)$$

$$a_{i;j} = \begin{cases} 1 & (j = i), \\ 0 & (j \neq i). \end{cases}$$

And in the case of $i \in \{m+1, \ldots, 2m\}$,

$$\mathbf{e}_i = -\mathbf{e}_{i-m}. \quad (14)$$

In the Step2, this paper calculates the derivative of the energy function with respect to $\mathbf{e}_i$ as follows:

$$E_{\mathbf{e}_i} = \frac{\partial E}{\partial \mathbf{e}_i} = \sum_{k=1}^{n-1} \sum_{l=k+1}^{n} \frac{\partial E_{k,l}}{\partial \mathbf{e}_i}, \quad (15)$$

where the derivative of $E_{k,l}$ is

$$\frac{\partial E_{k,l}}{\partial \mathbf{e}_i} = \frac{\ell_{k,l} - \rho(d_{k,l})}{1 - \rho(d_{k,l})} (\xi_{k;i} - \xi_{l;i})(\chi_k - \chi_l). \quad (16)$$

In the Step4, this paper employs the Newton-Raphson method for updating the current basis vectors. The modification vector $\Delta\mathbf{e}_i$ can be obtained as

$$\Delta\mathbf{e}_i = -\left[\frac{\partial^2 E}{\partial \mathbf{e}_i \partial \mathbf{e}_i^T}\right]^{-1} \frac{\partial E}{\partial \mathbf{e}_i}. \quad (17)$$

where $\mathbf{e}_i^T$ is the transposed vector of $\mathbf{e}_i$. The Hessian matrix can be expressed as

$$\frac{\partial^2 E}{\partial \mathbf{e}_i \partial \mathbf{e}_i^T} = \sum_{k=1}^{n-1} \sum_{l=k+1}^{n} f_{k,l,i}\mathbf{I} + g_{k,l,i}(\chi_k - \chi_l)(\chi_k - \chi_l)^T. \quad (18)$$

where $\mathbf{I}$ is the identity matrix in the m dimensional space, and $f_{k,l,i}, g_{k,l,i}$ are defined as follows:

$$f_{k,l,i} = \frac{\ell_{k,l} - \rho(d_{k,l})}{1 - \rho(d_{k,l})} (\xi_{k;i} - \xi_{l;i})^2,$$
$$g_{k,l,i} = \frac{(1 - \ell_{k,l})\rho(d_{k,l})}{(1 - \rho(d_{k,l}))^2} (\xi_{k;i} - \xi_{l;i})^2. \quad (19)$$

D. Pattern Recognition based on Space Folding Model

This section applies the space folding model to classification problem. Fig 2 shows the flow of multi-class classification based on the space folding model. First, the learning data set $\mathbf{X}_{train} = \{(\mathbf{x}_{train,i}, y_i), i = 1, \ldots, n_{train}\}$ is used to estimate the SFVs $\mathbf{e}_1, \ldots, \mathbf{e}_{2m}$. Second, this paper transforms the coordinate of the $\mathbf{X}_{train}$ into $\chi_{\mathbf{train}}$ by using Eq. 8. Next, a learning machine is trained with X_{train}. This paper employs Linear Discriminant Analysis (LDA)[11] or Neural Networks (NN) as the learning machines. This paper infers the label y^* of unknown unlabeled data set $\mathbf{X}_{test} = \{(\mathbf{x}_{test,i}, \text{unknown}), i = 1, \ldots, n_{test}\}$ by using the estimated SFVs and the learned model.

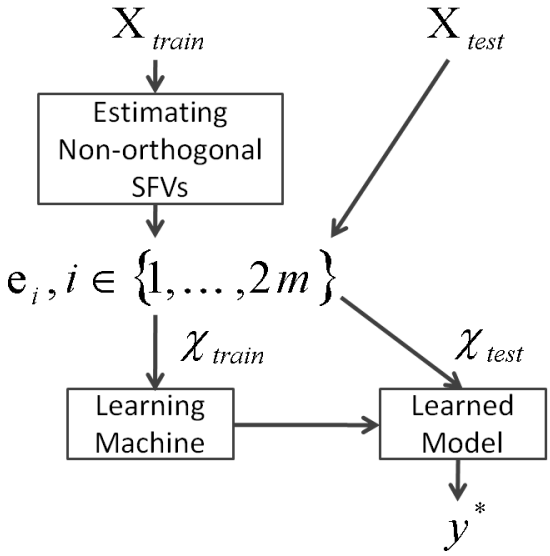

Fig. 2. Flow of classification based on space folding model.

III. Experiments and Discussion

A. Preliminary Experiment

This section examined the effectiveness of the proposed space folding model by using 2 kinds of data in 2-dimensional space. The first one is symmetric data. And the second one is non-symmetric data.

1) Symmetric Data: This paper uses 2 kinds of symmetric data, one of reflectively symmetric data and one of rotationally symmetric data. This paper constructs the space folding model, and learn the LDA for those data. The Fig. 3 shows the data distribution before and after constructing the model. The left of Fig. 3 shows the data distribution in the original space. The top left shows the reflectively symmetric data and the bottom left shows the rotationally symmetric data. The SFVs $\mathbf{e}_i, i = 1, \ldots, 4$ are shown by the filled '□', and the data of two classes are shown by the '○' and the '×'. The right shows the data distribution after the estimation of the non-orthogonal basis vectors. And it shows that all instances with the same label did gather when the model construction finished. Using the coordinates of data in the original space, the classification rate of the reflectively symmetric data was 71% and the classification rate of the rotationally symmetric data was 78%. And using the result of space folding model, the classification rates of the symmetric data were 96.5% and 98.5%.

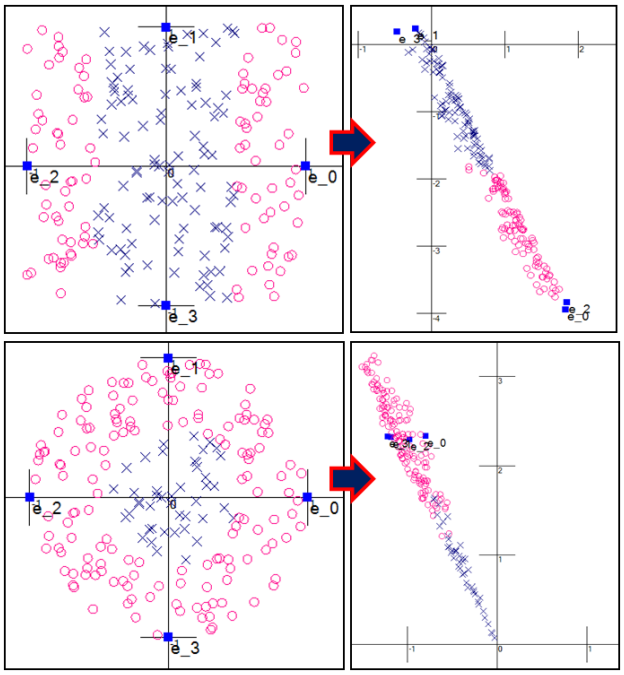

Fig. 3. The result of space folding model using symmetric data.

2) Non Symmetric Data: The paper uses a data distribution showed at the left figure of Fig 4. The right figure of Fig 4 shows the data distribution after the space folding in the case where the origin was used as the folding point. In this case, the data having the same label ('△') did not gather when the model construction finished. Then, this paper make the transformation of setting folding point at the center of mass and the SFVs by the eigen-vectors of the class where the eigen-value was the greatest among all classes. The Fig. 5 shows the data distribution before and after constructing the model based on this transformation. Because $\vec{e}_2$ got close to $\vec{e}_0$, the '△' data gathered to one group. And the classification rate of the proposed method using this transformation was 86% better than classification rate without this transformation (77.8%).

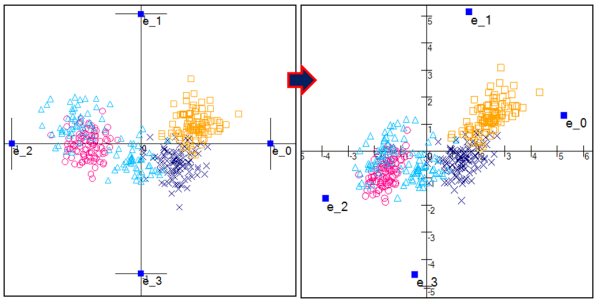

Fig. 4. The result of space folding model using non-symmetric data in the case where the origin was used as the folding point.

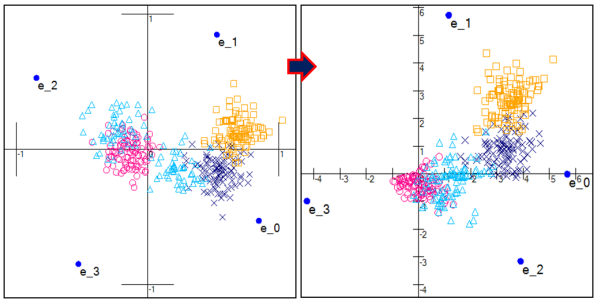

Fig. 5. The result of space folding model for non-symmetric data after setting the folding point at the center and the basis vectors by the eigen-vectors of the class which has the greatest eigen-value.

B. Hand-written Digits

This paper employed Pen-Based Recognition of Handwritten Digits dataset of the UCI Repository[10] as a two-class classification problem. The dataset consists of 10992 samples of 10 classes ('0' - '9') written by 44 people. This paper used 2110 samples out of 10992, which consist of digit '5' and digit '8'. It divided 671 samples written by 14 people into learning data D_1, and 1439 samples written by 30 people into test data D_2. In the collection of samples, the pen point coordinates with 100 msec intervals were measured on a tablet with a resolution of 500×500 pixels. Eight points $\{\mathbf{r}_l, l = 1, \cdots, 8\}$ dividing the orbit of the pen point into 7 equally long segments were chosen, and they were scaled to be that average became $\vec{0}$ and maximal variance along the horizontal or the vertical axes became 1. The aspect ratio was not changed in the scaling. Fig. 6 shows some examples of the handwritten digit '5', and Fig. 7 shows those of digit '8'. Though they show the handwritten trajectories, only the circled points were used for this experiment.

This section uses the learning data D_1 to learning the space folding model. This paper used 3 kind of learning machines (LDA, SVM and NN) to learn the classification model. Next this section uses the multi-dimensional scaling (MDS) method[12],[13] to visualize the learning data D_1 before and after the construction of the space folding model.

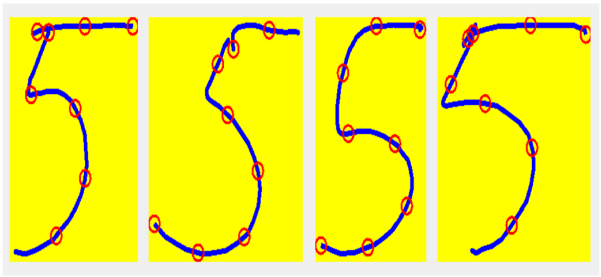

Fig. 6. Examples of hand-written digit '5'.

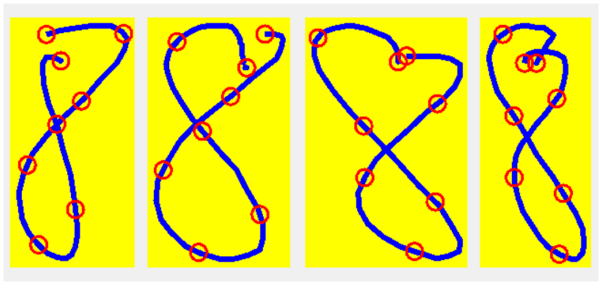

Fig. 7. Examples of hand-written digit '8'.

Fig. 8 shows the MDS result of the learning data D_1 before the construction of the space folding model. The data of digit '5' is shown by '$\times$', and digit '8' is shown by '$\circ$'. In Fig. 8, it is interesting that two groups of '5' were closer to '8' than the other '5' group, because some '5' were written in one continuous curve from upper-right to lower-left like digit '8'. Using the coordinates of original space, it is hard for linear learning machine classify correctly. Fig. 9 shows the MDS result of the learning data after the construction of the space folding model. Fig. 9 shows the data distribution of '5' and '8' gathering to each group and they are well-separated in the space folding model. So, it is easy for even a linear learning machine to classify.

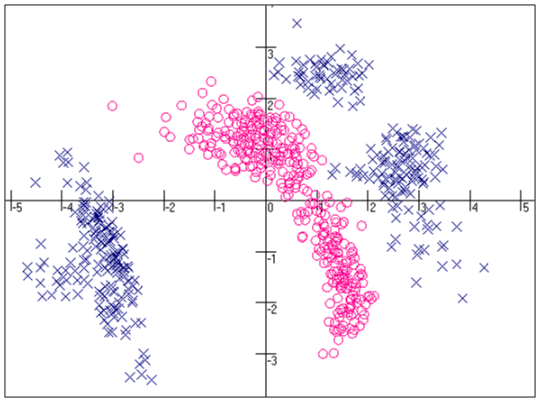

Fig. 8. MDS result of the learning data before the construction of the space folding model.

Then, this paper used the test data D_2 to evaluate the

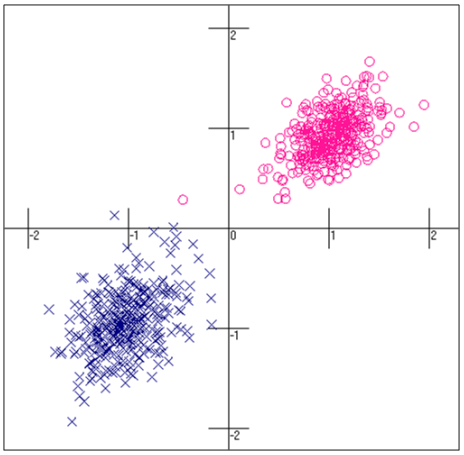

Fig. 9. MDS result of the learning data after the construction of the space folding model.

performance of 2-class classifications in 2 cases, with the space folding model (SFM) and without the SFM. Table I show the classification rate using LDA, SVM and NN. For the SVM, this paper used Gaussian kernel function $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\beta(\mathbf{x}-\mathbf{y})^2\right)$, where parameter $\beta = 1$ is the width of the Gaussian function. For the NN, this paper used 3-layer back-propagation network, and the type of activation function that has been used is the sigmoid function. To make a fair comparison, this paper set the hidden unit number $u_h = 4$ in the case of without SFM and the hidden unit number $u_h = 2$ in the case of with SFM. Table I showed that the accuracy of correct classification by the proposed method was better than the cases with only NN, LDA or SVM.

TABLE I
CLASSIFICATION RATE VIA LDA, SVM AND NN.

	Without SFM	With SFM
LDA	90.3%	97.3%
SVM	91.2%	99.2%
NN	96.3 ($\pm$1.3)%	98.5 ($\pm$0.7)%

IV. CONCLUSION

This paper proposed the space folding model which can change the similarity between the instances. The proposed method divided each basis vectors into positive and negative directions, and optimized $2m$ SFVs in the m dimensional space. It showed the algorithm to estimate the SFVs by minimizing the cross entropy energy function. Because the proposed method linearly transformed each quadrant differently from other quadrants and it could change the topology of data distribution, it could improve classification performance in comparison with the feature extraction by the conventional linear or nonlinear transformations with kernel methods. This paper also showed how to apply the space folding model to the classification problem. It showed the effectiveness of the proposed method through two experiments of classification. The preliminary experiment showed that the proposed space

folding model is effective for machine learning in the both cases of symmetric data and non-symmetric data. The numerical experiment applied the proposed method to classifying of a hand-written digits dataset of the UCI Machine Learning Repository. This experiment showed that the proposed method was effective for the machine learning. It showed that the accuracy of correct classification by the proposed method was better than the cases with only NN, LDA or SVM.

REFERENCES

[1] M. Aizerman, E. Braverman, and L. Rozonoer, Theoretical foundations of the potential function method in pattern recognition learning. Automation and Remote Control 25, pp. 821–837,1964.

[2] D. E. Rumelhart, G. E. Hinton and R. J. Williams, Learning representations by error propagation. In D. E. Rumelhart, J. L. McClelland and the PDP Research Group (Eds.), Parallel distributed processing (Vol. 1, pp. 318-362). Cambridge, MA: MIT Press, 1986

[3] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor, On kernel target alignment. Journal of Machine Learning Research, 2002.

[4] S. Wold, K. Esbensen, P. Geladi, Principal component analysis, Chemometrics and Intelligent Laboratory Systems, 2, pp. 37–52, 1987.

[5] R. A. Fisher, The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7, No. 2, pp. 179–188, 1936.

[6] B. Scholkopf, A. Smola and K. R. Muller, Kernel principal component analysis. Advances in Kernel Methods - SV Learning, pp. 327–352. MIT Press, Cambridge, MA, 1999.

[7] J. Arenas-Garcia, K. B. Petersen, and L. K. Hansen. Sparse kernel orthonormalized pls for feature extraction in large data sets. In Neural Information Processing Systems 19, 2006.

[8] C. E. Shannon, A mathematical theory of communication. Bell System Technical Journal 27, pp. 379–423, 1948.

[9] H. Theil, Economics and Information Theory. Rand McNally, 1967.

[10] A. Asuncion, and D. J. Newman, UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science, 2007.

[11] J.H. Friedman, Regularized Discriminant Analysis. Journal of the American Statistical Association, 1989.

[12] W. S. Torgerson, Theory and methods of scaling. New York, Wiley, 1958.

[13] A. Buja, D. F. Swayne, M. Littman, N. Dean, and H. Hofmann. XGvis, Interactive data visualization with multidimensional scaling. Journal of Computational and Graphical Statistics, 2001.