MVA2002 IAPR Workshop on Machine Vision Applications, Dec. 11 - 13, 2002, Nara - ken New Public Hall, Nara, Japan

11—1

A Multi-class Pattern Recognition Method by Combined Use of Multinomial Logit Model and K-Nearest Neighbor Rule

Osamu Hasegawa (1,2)* Takio Kurita (2)†

(1) Imaging Science and Engineering Laboratory
Tokyo Institute of Technology‡

(2) Neuro Science Research Institute
Advanced Industrial Science and Technology§

Abstract

In this paper, we propose a method for multi-class pattern classification by combined use of Multinomial Logit Model (MLM) and K-nearest neighbor rule (K-NN). Multinomial Logit Model (MLM) is one of the neural network models for multi-class pattern classification, and is supposed to be equal or better in classification performance than linear classification methods. K-NN is a simple but powerful non-parametric classification tool whose error probability does not exceed double of bayes error. However, it is also known that such high performance of K-NN is not always expected if number of dimension of input feature vector space is large. Therefore, first we train MLM using the training vectors, and then apply K-NN to the output of the MLM. By this, since K-NN is applied to the compressed low dimension vectors, it is expected not only to bring out natural performance of K-NN but also to shorten computation time.

Evaluation experiments were conducted by using some sets of non-artificial samples extracted from the handwritten character image database "ETL6". Those are (1) 36-classes (number + English capital letter), and (2) 82-classes (number + English capital letter + "Katakana" letter). Consequently, we obtained the following recognition rates: (1) 36-classes $\Rightarrow$ 100.0%, and (2) 82-classes $\Rightarrow$ 99.93%.

1 Introduction

In the research field of pattern recognition, classification of multi-class pattern is still on-going research issues.

Multinomial Logit Model (MLM) is one of the Generalized Linear Models [1], and is one of the neural network models for multi-class pattern classification. The classification performance of the model is supposed to be qeual or better than linear classification methods such as Linear Discriminant Analysis (LDA)[2].

K-nearest neightbour rule (K-NN) is known as one of the simplest but powerful non-parametric classification tools whose error probability does not exceed double of Bayes error[3]. However, it is also known that such high performance of K-NN is not always expected if number of dimension of input feature vector space is large[4]. Moreover, K-NN has the difficulty when number of class, samples and/or dimension of feature vector space are large, the amount of computation time becomes huge

Therefore, in this paper, we propose a method for multi-class pattern classification by combined use of MLM and K-NN. The outline of the proposed method is as follows. (1) train MLM using training vectors, (2) apply K-NN to the output (probability) vector of MLM. In the learning process of the MLM, we cooperatively applied 3 independent techniques those are thought to be useful to improve robustness of the network.

In this method, MLM functions as a pre-processor and reduces number of dimension of input vector, choosing the effective features for classification. Since K-NN is applied to the compressed low dimension vectors, it is expected not only to bring out natural performance of K-NN but also to shorten computation time.

We conducted evaluation experiments of this proposed method by using some sets of non-artificial samples extracted from the handwritten character image database "ETL6"[5]. Those are (1) 36-classes, 7200 test samples (number + English capital letter), and (2) 82-classes, 16400 test samples (number + English capital letter + "katakana" letter). The recognition rates were better than those when MLM or K-NN was applied to the data individually; (1) 36-classes $\Rightarrow$ 100.0% and (2) 82-classes $\Rightarrow$ 99.93% were obtained.

2 Multinomial Logit Model

2.1 Outline

Figure 1 shows a sample of a structure of MLM. We are given a set of training vectors $\boldsymbol{x}_i = (x_{i1}, ..., x_{iN})^T \in$

*E-mail: oh@ieee.org
†E-mail: takio-kurita@aist.go.jp
‡4259, Midori-ku, Nagatsuta, Yokohama, 226-8503 Japan
§1-1-1, Umezono, Tsukuba, 305-8566, Japan

R^N, and wish to train the network to be able to classify them into K classes $\{C_1, C_2, ..., C_K\}$. Teacher vectors for training are binary vector $\boldsymbol{t} = (t_1, ..., t_K)^T$ in which an element t_j that corresponds to the correct answer class C_j is 1, while the others are 0.

In this model, the output of the Kth output node is calculated as "softmax" of linear combination $\eta_k = \boldsymbol{a}_k^T \boldsymbol{x}$, as shown in the equations (1) and (2). Here, $\boldsymbol{x}$ and $\boldsymbol{a}$ denote input vector and parameter vector, respectively.

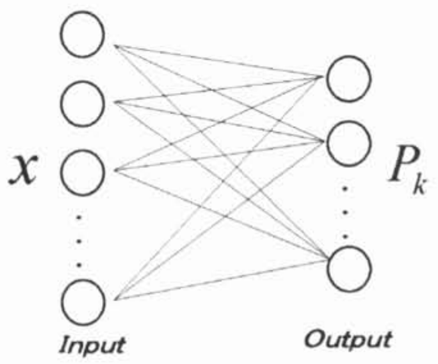

Figure 1: A Multinomial Logit Model

$$p_k = \frac{exp(\eta_k)}{1 + \sum_{m=1}^{K-1} exp(\eta_k)}, \quad k = 1, ..., K-1 \qquad (1)$$

$$p_K = \frac{1}{1 + \sum_{m=1}^{K-1} exp(\eta_k)}, \quad k = K \qquad (2)$$

Suppose parameter $\boldsymbol{A} = (a_1, ..., a_K)^T$ is a weight matrix of the network, a natural probability model of this network is given by

$$P(\boldsymbol{t}|\boldsymbol{x}; A) = \prod_{k=1}^{K} p_k^{t_k}. \qquad (3)$$

By taking the logarithm of the equation (3), the log likelihood $l(\boldsymbol{t}|\boldsymbol{x}; A) = logP(\boldsymbol{t}|\boldsymbol{x}; A)$ becomes

$$l(\boldsymbol{t}|\boldsymbol{x}; A) = \sum_{k=1}^{K-1} t_k \eta_k - log(1 + \sum_{m=1}^{K-1} exp(\eta_m)) \qquad (4)$$

The learning algorithm of this network can be obtained by using steepest descent method, and the gradient of this log likelihood is given as follows.

$$\frac{\partial l}{\partial \eta_k} = t_k - p_k, \qquad (5)$$

$$\frac{\partial \eta_k}{\partial \boldsymbol{a}_k} = \boldsymbol{x} \qquad (6)$$

$$\frac{\partial l}{\partial \boldsymbol{a}_k} = \frac{\partial l}{\partial \eta_k} \frac{\partial \eta_k}{\partial \boldsymbol{a}_k} = (t_k - p_k)\boldsymbol{x} \qquad (7)$$

Therefore, we obtain a simple weight update rule of the weigt $\{\boldsymbol{a_k}\}$ as

$$\boldsymbol{a_k} \Leftarrow \boldsymbol{a_k} + \alpha(t_k - p_k)\boldsymbol{x}, \quad (\alpha : learning\ rate) \qquad (8)$$

2.2 Cooperative application of 3 techniques for improving generalization ability

In this research, we cooperatively applied 3 independent techniques into the MLM learning process for improving generalization ability of the network. The techniques are proposed independently in the field of artificial neural networks.

2.2.1 Addition of noise

Kurita et al. showed that if noises are added to the hidden layer during training process, the network is structurized and it contributes to improve generalization error. This has the same effect to adding multiple "perturbed" images to samples for learning.

Therefore, in this study, we added random noise to the linear output $\eta_k = (\boldsymbol{a}_k^T \boldsymbol{x})$. ($\boldsymbol{x}$: input vector, $\boldsymbol{a}$: parameter vector.)

2.2.2 Tuning of learning coefficient based on entropy

In the training process, we made the network so that it selectively learns more about the data whose discriminant class are ambiguous. To put it concretely, we defined the "ambiguity of discernment class" by the entropy from the output of Equ.(1) as follows, and applied Equ.(9) to the Equ.(8).

$$\alpha\prime = \beta \times \sum_{i=1}^{K-1} (-p_i \times \log(p_i)) \qquad (9)$$

If classified discriminant class of an input data is ambiguous, the value of $\alpha\prime$ becomes large. We used this $\alpha\prime$ and the α in the Equ.(8). By this, it is expected that the decision boundaries become clearer.

2.2.3 Weight Decay

Hanson et al. proposed a method, called "Weight Decay"[8], which does not explicitly delete connections of few contributions in recognition performance but gradually brings them to 0. The Weight Decay can be summarized as a clause as follows.

$$-\lambda \boldsymbol{x} \qquad (10)$$

We applied the clause to the Equ.(8). By this, it is expected that the decision boundaries become clearer, too.

2.2.4 Modified updating rule

Consequently, Equ(8) is modified as follows.

$$\boldsymbol{a_k} \Leftarrow \boldsymbol{a_k} + (\alpha + \alpha')(t_k - p_k)\boldsymbol{x} - \lambda \boldsymbol{x} \qquad (11)$$

$$\alpha' = \beta \times \sum_{i=1}^{K-1} (-p_i \times log(p_i)), \qquad (12)$$

$$(\alpha, \beta, \lambda : learning\ rate)$$

We determined the optimal values of α, β and λ by the iterative simulations based on the vectors for training and cross-validation.

3 Feature Vector for Training and Test

3.1 Data for training and test

In this study, to evaluate the performance of the proposed method, we extracted some sets of samples from the handwritten character image database ETL6[5]. Those are

1. 36-classes (number + English capital letter) : 200 samples from each class : two sets of 7200 samples for training and test.
2. 82-classes (number + English capital letter + "Katakana" letter) : 200 samples from each class : two sets of 16400 samples for training and test.

Here, "number" denotes 10 kinds of letters from 0 to 9, "English capital letter" denotes 26 kinds of letters from A to Z, and "Katakana letter" denotes 46 kinds of Japanese letters. Figure 2 shows samples of "A"s extracted from the handwritten English capital letter images in the ETL6.

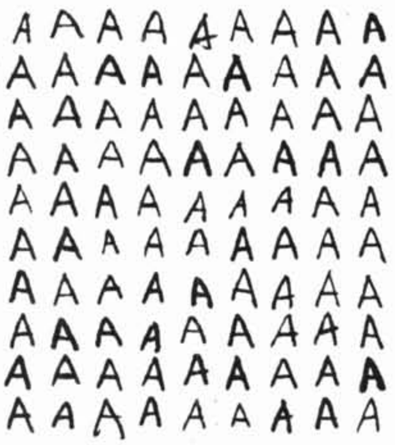

Figure 2: "A"s extracted from handwritten English capital letter images in ETL6

3.2 Basic feature vector

We first calculated four sets of "basic feature vectors" from the handwritten character images as follows. Two sets of them are the training samples for 36-classes and 82-classes, and the other two are the test samples.

1. Extract edges from each character image (30×30 pixel size) by Zero-crossing filter and remove noise.
2. Obtain four "4-directional edge feature images"[6] from each edge image, and resize them into 15×15 images.
3. Merge each set of the four 4-directional edge feature images into a 30×30 image.
4. Smooth all of the 30×30 images by 2-dimensional Gauss function.
5. Transform smoothed 30×30 images into 900×1 size vectors and regard them as the "basic feature vectors".

Since every class of samples contains 200 vectors, the total number of training or test vectors become (1) 32-classes: 7200 vectors, (2) 82-classes: 16400 vectors.

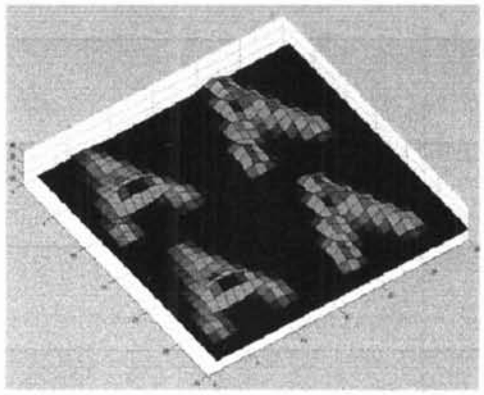

Figure 3: Sample of 30×30 pixel size image obtained from one of handwritten images of "A" in ETL6

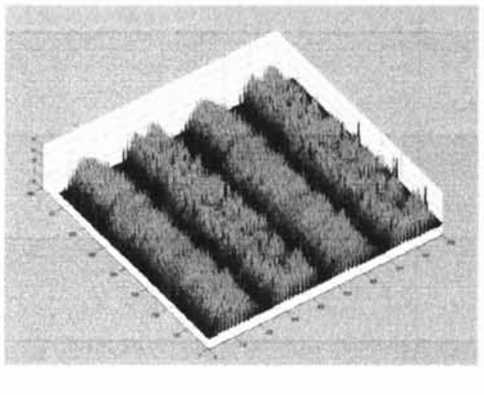

Figure 4: Visualized 7200×900 size matrix constituted from 7200 basic feature vectors.

Figure 3 shows a sample of a 30×30 pixel size image obtained from a handwritten character image of "A". In Fig.3, vertical direction features are shown in the right side column, and horizontal direction features are shown in the left side column.

Figure 4 shows a visualized 7200×900 size matrix constituted from the 7200 basic feature vectors for training. Each line in the direction of the horizontal axis corresponds to each character. Cyclic pattern shown in the figure corresponds to the 4-direction features respectively.

3.3 Kernel Feature Compound Vector

The basic feature vectors contain the image features.

On the other hand, in recent years Kernel method is getting popular, in which "Kernel feature vector" are constituted by Kernel functions. Since the kernel features seem to contain different kind of features from the basic feature vectors, more information for better performance is supposed to be obtained by compounding them.

Therefore, we secondly calculated two sets of "Kernel feature compound vectors" for training and test from the character images of 36-classes as follows.

1. Extract 100 basic feature vectors (3600 total) arbitrarily from each class of the 36-class basic feature vectors for training, and regard them as "representation vectors".

2. Calculate the Kernel feature vectors $\boldsymbol{y_i}^{train}$ and $\boldsymbol{y_i}^{test}$ from the representation vectors $\boldsymbol{x_k}$ and the input vectors $\boldsymbol{x_i}$ as follows.

$$\boldsymbol{x_k} = (x_{k1}, ..., x_{kN})^T,\ (N = 900) \qquad (13)$$

$$\boldsymbol{x_i} = (x_{i1}, ..., x_{iN})^T,\ (N = 900) \qquad (14)$$

$$\boldsymbol{y_i}^{train}, \boldsymbol{y_i}^{test} = (y_{i1}, ..., y_{ik})^T,\quad (i, k = 1, ..., 3600) \qquad (15)$$

$$y_{ik} = \exp\left(\frac{-\|\boldsymbol{x_k} - \boldsymbol{x_i}\|^2}{2 \times \sigma}\right),\quad (i, k = 1, ..., 3600) \qquad (16)$$

Here, $\boldsymbol{x_i}$ for the $\boldsymbol{y_i}^{train}$ is the same set of $\boldsymbol{x_k}$. For the $\boldsymbol{y_i}^{test}$, a set of $\boldsymbol{x_i}$ is extracted from the basic feature vectors of 36-classes for test (100 vectors from each class). By this, two sets of 3600 vectors of 3600-dimension are obtained for training and test, respectively.

3. Constitute the Kernel feature "compound" vectors by combining the basic feature vectors (900-dimension) and the corresponding Kernel feature vectors. By this, we obtain two sets of 3600 vectors of 4500-dimension (3600+900) for training and test, respectively.

Figure 5 shows example of the Kernel feature vectors for training. In Fig.5, $i = k$ of $\{\boldsymbol{x_i}, \boldsymbol{x_k}\}$ on the diagonal line. Since distance of them are 0, the values of the equation (16) become large.

Figure 6 shows example of the Kernel feature vectors for test. In Fig.6, the data near a diagonal line become large too, because distance of them are close to 0.

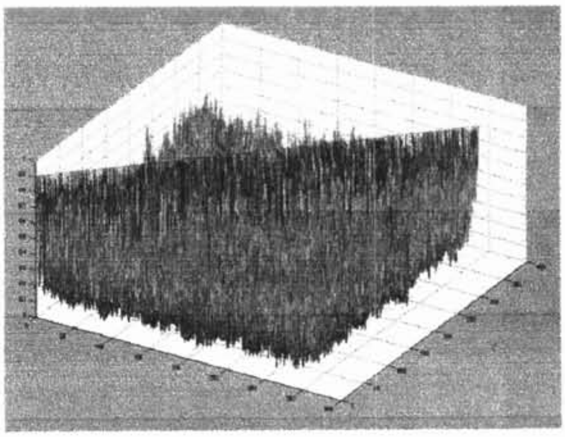

Figure 5: Examples of Kernel feature vectors for training.

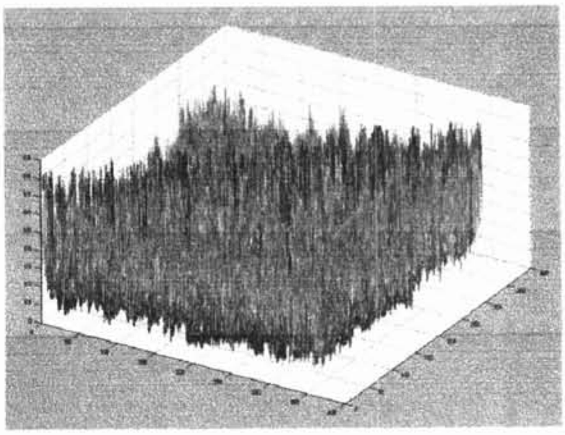

Figure 6: Examples of Kernel feature vectors for test.

4 Experiment and Result

4.1 Results by use of Multinomial Logit Model (MLM) Only

• Results by using Basic feature vector

Two MLMs were trained by use of the basic feature vectors (for training) of (1) 36-classes and (2) 82-classes. In training of the MLMs, enough time have spent till the discriminant rates for the training data become (1) 36-classes, more than 99.9 %, (2) 82-classes, more than 97.0 %

Subsequently, the outputs of the equations (1) and (2) are calculated by applying the sample vectors for test to the trained weight matrix of the network. Among them, the maximum value was compared with the correct answer as the class of recognition result. By this, we obtained the following results.

(1)36-classes: 94.86 %, **(2)82-classes: 92.97** %

• Results by using Kernel feature compound vector

The same processing was performed to the Kernel feature compound vectors of 36-classes, and obtained the following results.

36-classes: 97.89 %

As shown, the recognition rate was improved by use of the Kernel feature compound vector compared to the use of the basic feature vector. We consider that the same effect is expectable also to the data of 82 classes.

4.2 Results by Combined Use of MLM and K-NN

• Results by using Basic feature vector

Suppose the output of the equation (1) as vector $\boldsymbol{p} = (p_1, ..., p_{(K-1)})^T$.

$\boldsymbol{p_k}$ and $\boldsymbol{p_i}$ are calculated by applying $\boldsymbol{x_k}$ and $\boldsymbol{x_i}$ to the weight matrix of the MLM which was trained by the basic feature vectors (for training). K-NN are applied to the $\boldsymbol{p_k}$ and $\boldsymbol{p_i}$.

Here, in this study, we define distance between vectors $d_{i,j}$ as

$$d_{i,k} = \frac{1}{\|\boldsymbol{p_k} - \boldsymbol{p_i}\|} \tag{17}$$

(There might be other definitions.)

The numbers of test (unknown) samples are as follows; (1) 36-classes : 7200, and (2) 82-classes : 16400. Consequently, we obtained the following results.

(1) 36-classes: 99.98 %, **(2) 82-classes: 99.93** %

The dimension of the vectors are reduced by the MLM as shown below, before applying K-NN.

(1) 36-classes : 900 $\Rightarrow$ 36 dimension
(2) 82-classes : 900 $\Rightarrow$ 82 dimension

• Results by using Kernel feature compound vector

Similarly, we applied K-NN to the output vectors of the MLM traind by use of the Kernel feature compound vectors, and obtained the following results. The number of test (unknown) samples is 3600.

36-classes: 100.0 %

The dimension of the vector is drastically reduced by the MLM as shown below, before applying K-NN.

36-classes : 4500 dimension $\Rightarrow$ 36 dimension

5 Conclusion

In this paer, we proposed a method for multi-class pattern classification. The approach is based on the combined use of the multinomial logit model (MLM) and the "Kernel feature compound vectors". The Kernel feature compound vectors are compound feature vectors of the geometric image features and the Kernel features.

In the learning process of the MLM, we cooperatively applied 3 independent techniques for improving generalization ability of the network.

The MLM functions as a feature selector, that is, the MLM selects the effective features from very high-dimension vectors for classification.

Recently, Yasuda reported similar results to us, "the middle of 99%", from the same 36-classes data[7]. In his experiment, he extracted 12-direction feature field from the data and added perturbations (sifts, scale changes and rotations) to them. For recognition, he used correlation method. That is, we consider that by the proposed method in this paper, we can obtain similar (or better) results in condition of fewer samples and less information than his method.

Detailed performance comparison with other technology such as Support Vector Machines is one of the next research topics.

References

[1] P.McCullagh and J.A.Nelder FRS: Generalized Linear Models, Chapman and Hall (1983)

[2] O.Duda et al.: Pattern Classification, John Wiley & Sons, Inc., (2001)

[3] T.Cover and P.Hart : "Nearest Neighbour Pattern Classification", IEEE Trans. Inf. Theory, Vol.IT-13, No.1, pp.1-27, (1967)

[4] K.Fukunaga : "Bias of Nearest Neighbour Error Estimation", IEEE Trans. PAMI, Vol.PAMI-9, No.1, pp.103-112, (1987)

[5] Electrotechnical Laboratory (ETL) Character Database, (Contact : etlcdb@m.aist.go.jp), http://www.etl.go.jp/~ etlcdb/#English

[6] M.Yasuda, H.Yamada, T.Saito, K.Yamamoto, and T.Hananoi: "An improved correlation method for character recognition - A proposal of reciprocal feature extraction", J. of Systems Computers Controls, Vol.15, No.4, pp.29-38, (1984)

[7] M.Yasuda : "Effect of Perturbations for Correlation Method", Technical Report R01-4-4, Committee for Data Input System by Using Image Recognition Techniques, 2001, (in Japanese)

[8] Hanson,S.J. and Pratt,L.Y.: "Comparing Biases for Minimal Network Construction with Back-Propagation", in Touretzky,D.S. ed., Advances in Neural Information Processing Systems 1, pp.177-185, Morgan Kaufman, 1989