Pattern Recognition by Distributed Coding: Test and Analysis of the Power Space Similarity Method

Takao KOBAYASHI† Masaki NAKAGAWA‡
†‡ Graduate School of Technology, Tokyo University of Agriculture and Technology
2-24-16 Nakacho, Koganei-shi, Tokyo, 184-8588 Japan
E-mail: † kobayashi@hands.ei.tuat.ac.jp, ‡ nakagawa@cc.tuat.ac.jp

Abstract

This paper considers pattern recognition methods using distributed coding. These methods permit rapid learning from a large number of training samples; their recognition speed is high regardless of the size of the learning samples. This paper presents both basic algorithm and extended algorithms. Experiments with a large database of off-line handwritten numeric patterns are then described using the power space similarity method, being a type of distributed coding. Finally the effectiveness of the technique is considered.

1. Introduction

In any particular feature extraction method, the number of learning samples can be considered to be a basic factor determining the performance of pattern recognition and learning methods. The fact that the larger the number of samples, the higher the recognition rate becomes, applies to the majority of learning methods. In the case of the k-NN method, for example, the recognition rate monotonically increases with the number of samples unless patterns are inseparable in the feature space.

Increasing the number of learning samples, however, gives rise to several problems.

With some recognition algorithms, the recognition speed drops significantly as the number of learning samples increases. It is known that this is a critical problem with the NN method [1].

With some other recognition algorithms, the learning process takes an extremely long time. To decrease the time, this type of algorithms deals with learning as an optimization problem, and uses various techniques to efficiently search for solutions with the smallest effort. If a large number of samples and categories are involved, however, such algorithms sometimes fail to give proper solutions.

Then, we have a question that:

"Given a certain learning or recognition method that can learn and recognize patterns with high speeds, is it possible to increase the recognition rate as more learning patterns are provided?"

We consider this question by proposing a learning/recognition model, present experiments and analyze the results. We call the model as "distributed coding", whose time complexity for recognition is constant and independent of the number of learning samples. In the case of the simplest model of this type, the learning time is proportional to the number of learning samples.

This paper describes the relationships between the number of learning samples and the recognition rates that have been verified using a database of off-line handwritten character patterns. The recognition rate using the 1-NN method was measured under the same learning conditions to compare the performance.

2. Basic Principles of the Pattern Recognition by Distributed Coding

2.1. Distributed Representation of Vectors

We assume that the sizes of any pattern feature vector are equal. Therefore the feature space is subset of N-dimensional hypersphere having its origin in the center.

Let $\boldsymbol{x}$ be a feature vector. Space Q is defined as a discrete hyperspheric space. A number, p, of the points close to $\boldsymbol{x}$ in Q are selected. These points are called p nearest neighbor points to $\boldsymbol{x}$ in Q, and are expressed as $Q(\boldsymbol{x}, p)$.

Multiple mutually different spaces can be considered. If N_Q spaces , Q_0, Q_1,…, Q_{NQ-1}, are used to create Q_k ($\boldsymbol{x}$, p_k), then $\sum p_k$ points are selected.

Now let $N_e = \left|\sum Q_k(x, p_k)\right|$, and denote $\{e_j\} \equiv \cup Q_k(x, p_k)$ $(j = 0,\ldots, N_e - 1)$. We call $\{\boldsymbol{e}_j\}$ "distributed representation of $\boldsymbol{x}$." $\{\boldsymbol{e}_j\}$ includes the information of $\boldsymbol{x}$.

We are concerned about such a case as

$$\boldsymbol{x} \cong \alpha \sum \boldsymbol{e}_j \qquad (1)$$

and Q, p that suffice above expression. (Note that the coefficient α has no sense because vectors $\boldsymbol{e}_j$ are in hypersphere.)

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

2.2. Mathematical background

We introduce mathematical background for pattern recognition by distributed representation .

Let S be a N-dimensional hyperspherical space. Choose one vector $\boldsymbol{v}$ in S.

Let $W_N(\theta) = \{u \mid (u,v)/|u|\cdot|v| \le \theta, u \in S\}$. Then $W_N(\pi) = S$.

The value of $W_N(\theta)/W_N(\pi)$ ranges from 0 to 1, and grows larger and larger as θ grows. For larger N, it rises sharply when θ is near $\pi/2$. This fact is called as "Tendency to Orthogonality" [2].

Given points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, a value of $|(\cup Q_k(x_1,p_k) \cap (\cup Q_k(x_2,p_k))|$ can be a measure for similarity. From the tendency to orthogonality in high dimensional spaces, the value is more reliable when the distance between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is near.

Based on this fact, we can constitute such a data structure for pattern recognition that there always exist learning patterns close to an unknown input pattern.

If points are distributed randomly in the space, the probability that points exist in certain local region depends on the average density in the region. If the distribution is dense, there exist points statistically stably.

Thus, we can take a strategy that a large number of learning samples should be prepared in order to have sufficient density.

2.3. Power Space Similarity Method

To employ the distributed representation, the original feature vector space can be any space as far as it is a hypersphere, and the space Q and the number p can be selected in numerous different ways.

In this paper, we adopt a method that we call "power space similarity method" [3]. It is a distributed representation defined as follows:

$$x \in N!^{\pm},\ Q_k = {}_N C_1^{\pm}, \ldots, {}_N C_{N-1}^{\pm},$$

where,

$$N!^{\pm} \equiv \{x = (x_0, x_1, \cdots, x_{N-1}) \mid x_i \in \{2k - N + 1\}(k = 0, \cdots, N-1), x_i \neq x_j (i \neq j)\}$$

$${}_N C_k^{\pm} \equiv \{x \mid x \in \{-1,+1\}^N, |x| = 2k - N\}$$

The nearest neighbor points $\boldsymbol{x}_0$, $\boldsymbol{x}_1$, ... , $\boldsymbol{x}_{N-1}$ from $\boldsymbol{x}$ to ${}_N C_1^{\pm}, {}_N C_2^{\pm}, \ldots, {}_N C_{N-1}^{\pm}$ are obtained respectively.

A basic algorithm for pattern recognition using the distributed representation can now be discussed. Because $Q(\boldsymbol{x}, p)$ is a subset of Q, it is the element of the powerset 2^Q. Namely, $Q(\boldsymbol{x}, p)$ can be expressed as an element of $\{0,1\}^{|Q|}$. If such binary vectors are created from spaces Q_k (there are N_Q spaces), a very high-dimensional binary vector can be created by arraying all these vector elements. This binary vector is expressed as follows:

$$\boldsymbol{b} = (b_i) \in \{0,1\}^{\sum |Q_k|}.$$

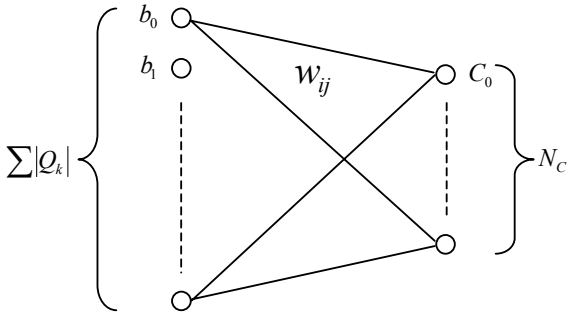


Figure 1. High-dimensional single-layer discriminant function model shown in the form of a distributed representation.

This is to map $\boldsymbol{x}$ into a very high-dimensional space. When $\boldsymbol{x}$ is mapped into a very high-dimensional space whose dimension is higher than the number of samples, it always becomes linearly separable. Therefore, it is possible to build a linear discriminant function with the input layer of $\sum |Q_k|$ and the output layer of N_C elements (See figure 1.). The category is defined as C_j (j=0, … , N_C -1).

To recognize an unknown pattern $\boldsymbol{x}$ $\boldsymbol{x}$ is converted to $\boldsymbol{b}$, and the following equation is calculated:

$$score(x, C_j) = \sum b_i w_{ij} \qquad (2)$$

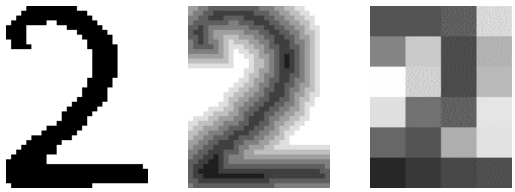
Category C_k with satisfying $k = argmax\{score(\boldsymbol{x}, C_j)\}$ is the result of recognition.

score( ) is equivalent to similarity. In this structure, the weight coefficient w_{ij} ,that allows learning samples to be recognized completely can be determined by performing the following steps:

Set all w_{ij} to 0. Convert a learning sample $\boldsymbol{x}$ that belongs to category C_i to $\boldsymbol{b}$. If the element b_i of $\boldsymbol{b}$ is 1, set w_{ij} to 1. Perform this procedure on all learning samples. Full marks are always obtained regarding learned patterns. Therefore, right answers are always obtained except a case in which full marks are obtained to two or more categories.

The dictionary only has to memorize information on i, j of w_{ij} = 1. Calculation of the score only require to judge whetherr w_{ij} is 0 or 1 for such i as b_i = 1. If the number of learning samples is defined as N_L, there exists an algorithm such that the learning time for the dictionary $O(N_e \times N_L)$ and its recognition time is $O(N_e)$. The size of this dictionary is not more than $O(N_e \times N_L)$. That is, the learning time is proportional to the number of learning

samples, and the recognition time is constant, independently of the dictionary size.



(a) (b) (c)

Figure 2. Preparation of the feature vector for a handwritten numeral pattern.

(1) Quantization of a real value vector to a vector in *N!*

$$v = (120,150,300,490,710) \in R^6$$

$$\mapsto \quad x = (-5,-3,-1,+1,+3,+5) \in 6!$$

(2) Distributed representation of *x*

$$e_1 = (-1,-1,-1,-1,-1,+1) \in_6 C_1^{\pm}$$

$$e_2 = (-1,-1,-1,-1,+1,+1) \in_6 C_2^{\pm}$$

$$e_3 = (-1,-1,-1,+1,+1,+1) \in_6 C_3^{\pm}$$

$$e_4 = (-1,-1,+1,+1,+1,+1) \in_6 C_4^{\pm}$$

$$e_5 = (-1,+1,+1,+1,+1,+1) \in_6 C_5^{\pm}$$

$$x = \sum_{j=1}^{5} e_j$$

(3) Mapping to binary vector space

$e_1 \in \{$(-1,-1,-1,-1,-1,+1), …,(+1,-1,-1,-1,-1,-1)$\}$

$e_2 \in \{$(-1,-1,-1,-1,+1,+1), (-1,-1,-1,+1,-1,+1), …, (+1,+1,-1,-1,-1,-1)$\}$

….

b=(1000001000000000000001000000000000000000010 0000000000000100000) $\in \{0,1\}^{\left|\sum_6 C_k^{\pm}\right|}$

Figure 3. A numerical example of distributed coding.

3. Recognition Experiment

3.1. A Basic Method for Handwritten Numerals

The relationships between the number of learning samples and the recognition rates were studied using handwritten numeral patterns.

This database contains 19,000 samples for each of ten numerals. The total number of samples is 190,000. The database contains so many samples that it is suitable for experiment in this paper.

Figure 2 shows how a feature vector is created. The original pattern is a binary image as shown in figure 2(a). The original pattern is eroded gradually and it is also dilated and all the patterns are then superimposed to create a multi-valued pattern as shown in figure 2(b). The pattern is then divided into small grids of equal size, the number of which is defined as N. The sum of pixel values from each grid is arrayed to form a vector as shown in figure 2(c).

Next, an N-dimensional real number vector created in this way is quantized to a vector in *N*!. Thus the feature vector is obtained. Then, it is expressed as a distributed representation, as explained in section 2.3. Figure 3 shows a numerical example applying distributed coding. (Although 6-dimensional vector space is used in the figure, larger dimension is used in practice.)

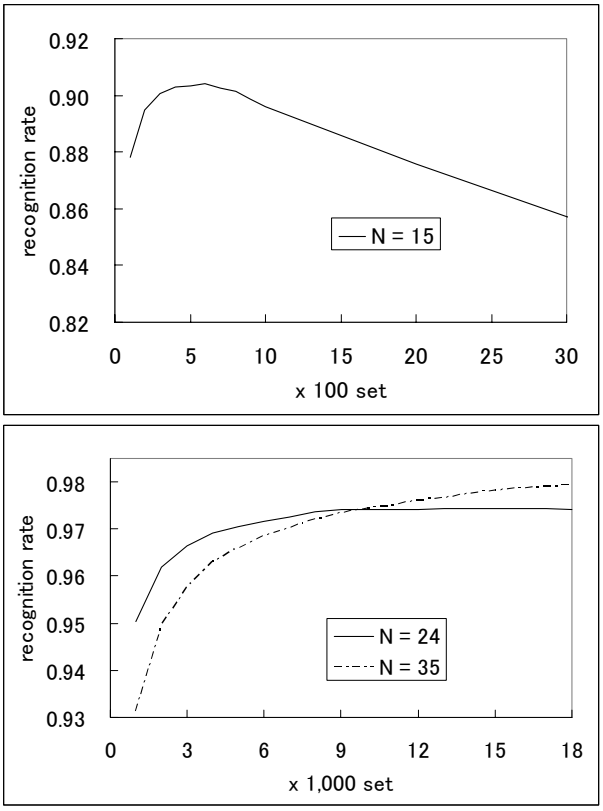


Figure 4. Relationships between the number of learning samples and the recognition rate.

We tested the cases when the numeral pattern is partitioned into 3x5, 4x6 or 5x7. So the dimensions of feature vectors are 15, 24, or 35 respectively. Figure 4 shows the results.

For N=15, learning saturation has occurred since the dimension is too small. When N=24, saturation is about to occur as the number of learning samples increases. When N=35, saturation is not observed with the given

number of samples. If the number of samples is increased, the recognition rate seems to improve further. To differentiate it from the improved method, which is mentioned below, the basic algorithm presented in section 2.3 is called "type-0".

3.2. Improved Weight Coefficient

Because the structure of this method is characterized by the high-dimensional linear discriminant function shown by equation (2), we can adjust the weight coefficient w. In this section, we take the approach called "ensemble learning" [4].

It is here assumed that an adequate number of samples exist according to the true distributions. Samples, the number of which is defined as N_L, are drawn randomly to make a dictionary. By repeating the process of drawing N_L samples, we can obtain a multiple number of dictionaries. Each of these dictionaries is used to recognize an input pattern, and the total of scores given by all the dictionaries is considered in the final result of recognition. The recognition rate is usually better than that obtained by using only a single dictionary.

If a pattern $\boldsymbol{x}$ is taken randomly from a category C to make a dictionary and if $\boldsymbol{x}$ is converted to $\boldsymbol{b} = (b_i)$, we have the equation shown below, provided that the probability that w_{ij} is equal to 1 is p_{ij}:

$p_{ij} = \Pr(b_i=1 \text{ and } C=C_j)$.

This ultimately leads to one linear discriminant function that has $1-(1-p_{ij})^{N_L}$ as its weight coefficient w_{ij} (see Figure 5).

This makes it possible to achieve the same recognition speed as the type-0 algorithm without increasing the memory area.

In practice, however, a true probability of occurrence p_{ij} cannot be obtained. Because an approximate value $\hat{p}_{ij}$ can be obtained from a group of samples, this approximate value must be used. To determine an optimum N_L, test patterns must be prepared in addition to learning samples. The value N_L that allows the recognition rate to reach a maximum value is adopted.

Figure 6 (a) shows the recognition rates measured in the experiment with N=24, where the same learning samples were used, as mentioned in section 3.1. It is evident that the recognition rates obtained using this approach are better than that obtained using type-0. The recognition rates obtained using the 1-NN method are also shown for the purposes of comparison.

If the results are compared, based on the same number of learning samples, the recognition rates obtained using the NN method are better than those obtained using other methods.

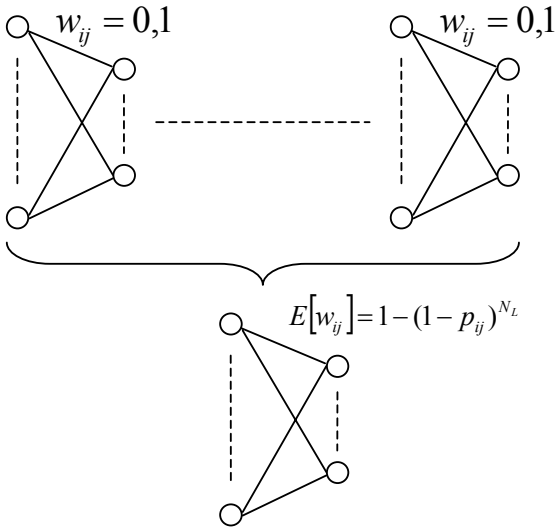
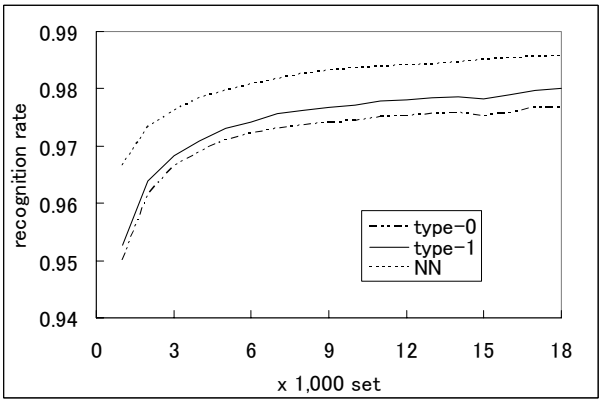
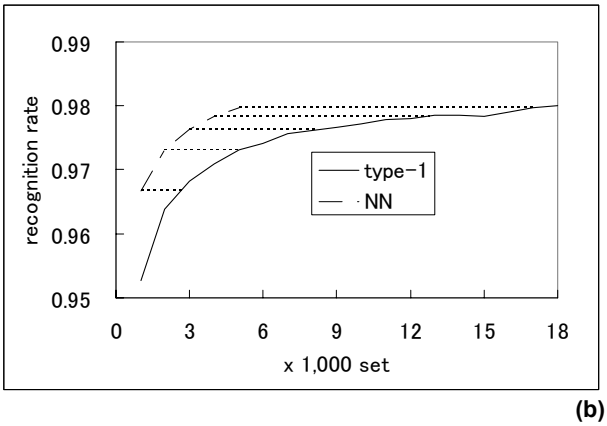


Figure 5. Dictionary structure with a score equivalent to the average score given by many dictionaries.



(a)



(b)

Figure 6. Comparison of the recognition rates by type-0, type-1, and the NN method.

If a larger number of samples are used when employing this method, the resultant recognition rates can be improved to the level of those obtained using the NN method. Figure 6 (b) shows that the recognition rates obtained using the NN method with employing 1,000 to 5,000 sets are equivalent to those obtained using the type-1 with 3,000 to 18,000 sets.

This structure adjusts the weight coefficients with probability-of-occurrence functions and is defined as the "type-1" algorithm.

3.3. Further Improvement

Another feasible approach is to upgrade the learning function by adjusting the weight coefficients for wrong learning patterns, considering the distributions of each category.

Based on this approach, an attempt was made to optimize the weight coefficients by using AdaBoost [5] [6], a typical method of upgrading learning functions.

In measuring recognition rates using this approach, the same experimental patterns were used, N was set to 24, and 9,000 sets were used for learning, with various learning parameters being tried. Using a particular set of parameters, the recognition rate was 97.85%. It was verified, therefore, that the recognition performance of this method could be increased to be greater than that of type-0 and type-1.

A method of successively updating the weight coefficients in some manner is defined as the "type-X" algorithm.

Table 1. Recognition rates of each method with N=24 and 9000 set samples.

	Recognition rate
type-0	0.9741
type-1	0.9767
type-X	0.9785
NN method	0.9830

3.4. Speed

In the recognition process using distributed coding, the value of w_{ij} is searched for in a dictionary of $b_i = 1$. That is, the recognition process is completed by performing the search and then performing additions of $(N-1)\times N_C$ times.

Using the 1-NN method, inner products generated from all learning samples must be calculated. Therefore, the recognition process is completed by performing multiplications and additions of $N_L \times N$ times. To increase the speed of the NN method, various techniques are proposed (for example, [7]). The NN method, however, calculates the similarity between all the categories and is thus not the type of method designed to increase speeds at the expense of accuracy. Moreover, it has the limitation that the learning time is $O(N_L)$, and therefore the number of calculations cannot be decreased.

Table 2 shows the measured recognition time. In the case of the methods proposed in this paper, the recognition time is quick and the number of learning samples does not affect the recognition speed significantly. On condition of the same recognition rate, discrimination speed of type-0, 1, and X can become about 100 or 1,000 times as fast as the 1-NN method.

Table 2. Recognition time per character (Pentium III, 800 MHz).

(milli sec.)

		1-NN	*4 type-0	*3 *4 type-1,X
Feature extraction process (A) *1 *2		0.063	0.063	0.063
Discrimination process (B)				
	10,000 samples	3.1	0.009	0.023
	30,000 samples	8.9	0.009	0.024
	50,000 samples	14.8	0.009	0.025
	90,000 samples	26.7	0.009	0.027
	180,000 samples	53.4	0.009	0.029

*1 recognition time = feature extraction time (A) + discrimination time (B)

*2 feature extraction time of each method is the same.

*3 recognition time of type-1 and type-X are theoretically the same.

*4 discrimination time of distributed coding may vary by implementation. When employing a hash table, it depends on the table size.

4. Conclusion

A series of pattern recognition algorithms based on distributed coding were proposed and tested. Characteristics of these algorithms are summarized as follows:

The dictionary has a linear discriminant structure, permitting a very fast recognition speed. Similarity can be calculated for all the categories. The algorithm can be applied to classification problems for a large category set.

Type-0 : Learning can be completed by referencing all the learning samples only once. The dictionary has a linear discriminant structure, and the weight coefficients are 0 or1.

Type-1 : The learning speed is fast. The weight coefficients are determined from frequencies of occurrence and have real values.

Type-X : The weight coefficient can be optimized by successive learning. The weight coefficients have real values.

Recognition experiments were conducted using real patterns, and the relationship between the number of learning samples and the recognition rates were examined. When compared based on the same number of learning samples, it was found that although the recognition rate of the new method was initially lower than that of the NN method, it is possible to increase the recognition performance to the level of that of the NN method.

The recognition speed is not affected by the number of learning samples, and is very fast. This shows that a high-speed recognition algorithm can be trained to increase its recognition rates by employing a large number of training samples.

This study was made with a focus on the adjustment of weight coefficients using the power space similarity method chosen from various distributed coding methods. By making further modifications or improvements, for example, by modifying the distributed representation method, there is the possibility that a recognition rate equivalent to that of the NN method can be achieved.

Acknowledgments

Deep appreciation is expressed to BIRDS Systems Research Institute, Inc. for their provision of the database for character recognition.

References

[1] T.M. Cover and P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Information Theory, vol.IT-13, no1, pp.21-27, Jan. 1967

[2] P. Kanerva, Sparse Distributed Memory, The MIT Press, Cambridge, Massachusetts 1988.

[3] T, Kobayashi, Method of and apparatus for pattern recognition and method of creating pattern recognition dictionary, United States Patent, 5,689,584, Nov. 1997.

[4] L. Breiman, Bagging predictors, Machine Learning, vol.24, no. 2, pp.123-140, 1996.

[5] Y. Freund and R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences, 55, pp.119-139, 1997.

[6] N. Murata, Geometrical Understanding of Boosting Algorithms, (in Japanese) Technical Report of IEICE, PRMU2002-97, NC2002-50, pp.37-42, Oct.2002.

[7] T. Shibata, T. Kato and T. Wada, K-D Decision Tree, (in Japanese) Technical Report of IEICE, PRMU, Vol.103, No.295, pp.85-90, Sep.2003.