RECOGNITION OF HAND-WRITTEN PATTERNS BY ROTATION-INVARIANT NEOCOGNITRON

Shunji Satoh†, Jousuke Kuroiwa¶, Hirotomo Aso† and Shogo Miyake‡
Email:shun@aso.ecei.tohoku.ac.jp

† Department of Electrical Communications, Tohoku University, Sendai, 980-8579, Japan.
¶ The Division of Mathematical and Information Sciences, Hiroshima University, 739-8521, Japan.
‡ Department of Applied Physics, Tohoku University, Sendai, 980-8579, Japan.

ABSTRACT

A rotation-invariant neocognitron, which has been recently proposed by authors, is trained by using hand-written numerical patterns provided by ETL-1 database. A new learning algorithm, "auto-generating algorithm", is proposed and the learning time is reduced. The model can recognize realistic hand-written patterns. It is also shown that the model is completely robust for rotations of hand-written patterns.
KEYWORDS: neocognitron, ETL-1, rotated pattern

1. INTRODUCTION

A neocognitron [1], which is a hierarchical neural networks for pattern or signal recognition, is considerably robust for distortion, scaling, and/or translation of patterns but it can not recognize patterns which are rotated in large angles. A rotation-invariant neocognitron was proposed by authors in order to make up the weak point of the original neocognitron [2],[3]. In the new model rotational information of inputs are added besides positional information in the original neocognitron. The rotation-invariant neocognitron could recognize globally and/or locally rotated patterns as well as translated, scaled and distorted ones. In the simulation, we used patterns produced by use of a computer.

In this paper, we examine whether the rotation-invariant neocognitron can recognize realistic hand-written patterns. For this purpose, we make use of a numbers of hand-written numerical patterns given in ETL-1 database[1]. In practice, various patterns more than one thousands are used to test the model. In constructing a model a new method , "auto-generating algorithm", is proposed for shortening learning time.

2. ROTATION-INVARIANT NEOCOGNITRON

In this section, a rotation-invariant neocognitron is briefly reviewed.

2.1. Structure

A rotation-invariant neocognitron is composed of cascaded connections of a number of modular structures U_{Sl} and U_{Cl} preceded by an input layer U_0 as shown in Figure 1. Here U_{Sl} denotes a layer consisting of S-cells in the lth module, and U_{Cl} a layer consisting of C-cells. A layer is composed of a number of cell-plane stacks, and different cell-plane stacks detect different features of inputs (a cell-plane stack is referred to as "CPS" hereafter). A CPS is composed of a number of cell-planes [1], and each cell-plane detects a different rotation angle of the features. Each cell in a CPS is located in a three-dimensional space. In this model a rotational information of an input pattern is represented by the number n_θ assigned to a cell-plane in a CPS, and a positional information by a position (n_x, n_y) of a firing cell in the specific cell-plane.

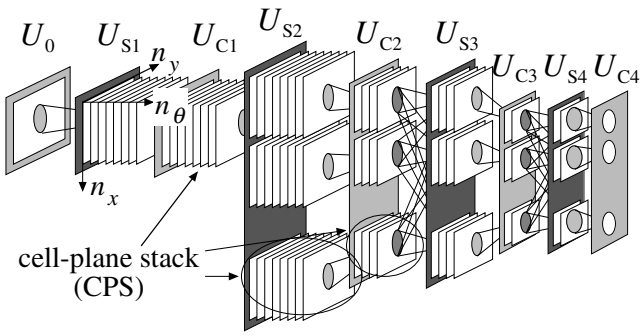

Figure 1: A structure of rotation-invariant neocognitron.

An output response of an S-cell located on $\boldsymbol{n} = (n_x, n_y, n_\theta)$ of the kth CPS in the lth module is denoted by $u_{Sl}(\boldsymbol{n}, k)$, and an output response of a C-cell by $u_{Cl}(\boldsymbol{n}, k)$. The CPS with serial number k is denoted as CPS#k. The output response $u_{Sl}(\boldsymbol{n}, k)$ is given by [1]

$$u_{Sl}(\boldsymbol{n}, k) = r_l \cdot \phi\left[\frac{1+e}{1+\frac{r_l}{1+r_l}\cdot i} - 1\right], \tag{1}$$

$$\phi(x) = \max(x, 0), \tag{2}$$

where

$$e = \sum_{\kappa=0}^{K_{Cl-1}-1} \sum_{\boldsymbol{\nu}\in A_l} a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, k) \cdot u_{Cl-1}(\boldsymbol{n} \underset{T_{Cl-1}}{\oplus} \boldsymbol{\nu}, \kappa), \tag{3}$$

[1] A character database published from the Electrotechnical Laboratory, Japan.

$$i \;=\; b_l(k)\cdot u_{\mathrm{V}l}(\boldsymbol{n}). \tag{4}$$

A binomial operator $\underset{M}{\oplus}$ with M is defined by

$$\begin{cases} n_x \underset{M}{\oplus} \nu_x \stackrel{\mathrm{def}}{=} n_x + \nu_x, \\ n_y \underset{M}{\oplus} \nu_y \stackrel{\mathrm{def}}{=} n_y + \nu_y, \\ n_\theta \underset{M}{\oplus} \nu_\theta \stackrel{\mathrm{def}}{=} (n_\theta + \nu_\theta) \bmod M. \end{cases} \tag{5}$$

Here r_l denotes a threshold value of an S-cell, $a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, k)$ represents an excitatory connection from a C-cell to an S-cell and $b_l(k)$ an inhibitory connection from a V-cell to an S-cell —a V-cell sends inhibitory inputs to S-cells and are not depicted in Figure 1 for simplicity. Each connection is linked to a restricted number of C-cells in the preceding module, $A_l(\subset \boldsymbol{Z}^3)$, K_{Cl} denotes the number of CPS's in the U_{Cl} layer, and T_{Cl} the number of cell-planes in the U_{Cl} layer.

An output response of a V-cell is given by

$$u_{\mathrm{V}l}(\boldsymbol{n}) \;=\; \sqrt{\sum_{\kappa=0}^{K_{Cl-1}-1} \sum_{\boldsymbol{\nu}\in A_l} c_l(\boldsymbol{\nu}) \cdot \left\{ u_{Cl-1}(\boldsymbol{n} \underset{T_{Cl-1}}{\oplus} \boldsymbol{\nu}, \kappa) \right\}^2} \tag{6}$$

where $c_l(\boldsymbol{\nu})$ is an excitatory connection from a C-cell to a V-cell, which takes a fixed value during a learning.

An output response of a C-cell is given by

$$u_{Cl}(\boldsymbol{n}, k) = \psi \left[\sum_{\boldsymbol{\nu}\in D_l} d_l(\boldsymbol{\nu}) \cdot u_{\mathrm{S}l}(\boldsymbol{n} \underset{T_{\mathrm{S}l}}{\oplus} \boldsymbol{\nu}, k) \right], \tag{7}$$

where the function ψ is defined by

$$\psi(x) = \frac{\phi(x)}{\phi(x)+1}. \tag{8}$$

Here $d_l(\boldsymbol{\nu})$ is an excitatory connection from an S-cell to a C-cell, $D_l(\subset \boldsymbol{Z}^3)$ represents a restricted region of the connection, and $T_{\mathrm{S}l}$ the number of cell-planes in the $U_{\mathrm{S}l}$ layer.

At a learning stage, excitatory connections $a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, k)$ and inhibitory connections $b_l(k)$ are modified.

2.2. Learning rule

A learning process can be divided into two stages. At the first stage a winner-cell (called as a *seed-cell*) of S-cells, which is denoted by $U_{\mathrm{S}l}(\hat{\boldsymbol{n}}, \hat{k})$, is chosen, and then excitatory connections $a_l(\boldsymbol{\nu}, \hat{\boldsymbol{n}}, \kappa, \hat{k})$ and inhibitory connections $b_l(\hat{k})$ are modified by an unsupervised learning given by

$$\Delta a_l(\boldsymbol{\nu}, \hat{\boldsymbol{n}}, \kappa, \hat{k}) \;=\; q_l \cdot c_l(\boldsymbol{\nu}) \cdot u_{Cl-1}(\hat{\boldsymbol{n}} + \boldsymbol{\nu}, \kappa), \tag{9}$$

$$\Delta b_l(\hat{k}) \;=\; q_l \cdot u_{\mathrm{V}l}(\hat{\boldsymbol{n}}), \tag{10}$$

where q_l is a learning constant in the lth module. At the next stage the other excitatory connections from C-cells to an S-cell at the $\boldsymbol{n} \neq \hat{\boldsymbol{n}}$ of the $k = \hat{k}$ cell-plane are modified by rotation-matrix method [3].

3. AUTO-GENERATING ALGORITHM

It is very important to adjust thresholds r_l of S-cells so that a neocognitron or a rotation-invariant neocognitron works with a high recognition rate. One prepares many thresholds for each layer, and then chooses an optimal threshold by observing firing rate and estimating a recognition rate for each threshold.

Let R_l be a single set of thresholds for testing in the lth layer ($R_l = \{r_l^{(1)}, r_l^{(2)}, \cdots\}$), N_{p} the number of training patterns, and t_{T} the time taken to train the model for a set of threshold and for a single training pattern. We shall now examine a simulation time. The learning time required to train the model for the all sets of threshold and for N_{p} training patterns is estimated by

$$t_{\mathrm{test}} = |R_2| \times \cdots \times |R_L| \times N_{\mathrm{p}} \times t_{\mathrm{T}}, \tag{11}$$

where $|\cdot|$ denotes the number of elements[2] and L is the number of a module. We use four modules in the model. If we use ten thresholds for each layer, five hundreds for N_{p} and assume ten seconds[3] for t_{T}, The total learning time t_{test} is estimated at about two months.

In order to reduce the learning time by a few orders of magnitude, we propose a new algorithm, "auto-generating algorithm", in this paper. In a conventional algorithm we train a lot of neocognitrons for all sets of threshold in order to produce the models corresponding to every set of threshold. In the new algorithm a single neocognitron is produced by training the model for only a single high thresholds and then many neocognitrons are generated from the single neocognitron by use of a new method, "CPS-unification method", in a very short time compared to $N_{\mathrm{p}} \times t_{\mathrm{T}}$.

A relation between firing rate of an S-cell and a threshold has been already analyzed in [5],[6]. An S-cell located at $\boldsymbol{n}$ of the kth CPS in the lth module has its own "weight vector" denoted by $\boldsymbol{a}_l(\boldsymbol{n}, k)$, and the S-cell receives outputs of C-cells in the preceding module. These outputs are referred to as an "input vector" denoted by $\boldsymbol{u}_{Cl-1}(\boldsymbol{n})$. The dimension of the weight vector is equal to that of the input vector, and is given by $|A_l| \times K_{Cl-1}$. A "similarity" between the weight vector and the input vector of an S-cell is defined as

$$s_l(\boldsymbol{n}, k) \stackrel{\mathrm{def}}{=} \frac{\boldsymbol{a}_l^{\mathrm{T}}(\boldsymbol{n}, k)\, \boldsymbol{u}_{Cl-1}(\boldsymbol{n})}{||\boldsymbol{a}_l(\boldsymbol{n}, k)|| \cdot ||\boldsymbol{u}_{Cl-1}(\boldsymbol{n})||}, \tag{12}$$

[2] $|R_1|$ is taken as 1 because an optimal r_1 can be easily chosen.
[3] t_{T} is estimated on SUN Ultra-Spark.

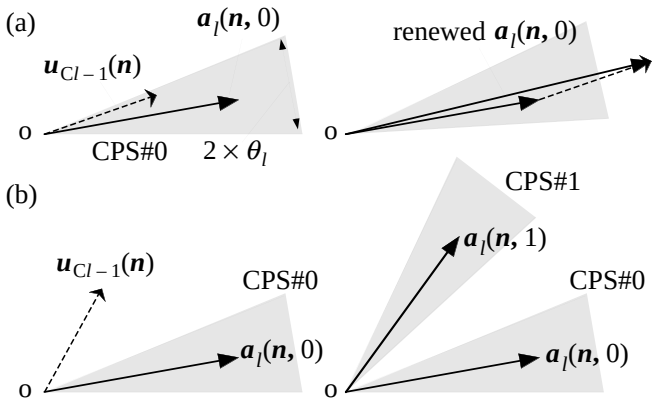

Figure 2: A learning process using a seed-cell. A gray region is an accept-region. (a): An input vector is trained by a learning of the S-cell in CPS#0. (b): An input vector is trained in new CPS#1 because the input vector is outside the accept-region.

and the firing rate of the S-cell is expressed as

$$u_{Sl}(\boldsymbol{n}, k) \simeq \phi\left[\frac{s_l(\boldsymbol{n}, k) - \tau_l}{1 - \tau_l}\right], \tag{13}$$

where $\tau_l = r_l/(1 + r_l)$. Equations (12) and (13) show that an S-cell in the lth module fires if and only if the similarity is larger than τ_l. This condition means that the angle between a weight vector and an input vector is smaller than an angle defined by

$$\theta_l \stackrel{\text{def}}{=} \cos^{-1}\tau_l = \cos^{-1}\frac{r_l}{1 + r_l}. \tag{14}$$

We call θ_l an "accept-region", which becomes narrower for a larger value of r_l.

Next, we briefly explain a learning method using a seed-cell selecting plane [4]. A seed-cell is generated in a seed-selecting plane (or a seed-selecting plane stack), and the position of each seed-cell, $\hat{\boldsymbol{n}}$, corresponds to the central position of a firing pattern in the previous layer. The firing pattern corresponding to $\hat{\boldsymbol{n}}$ is an important one that should be learned in the learning process. If S-cells at every $\hat{\boldsymbol{n}}$ in CPS's don't fire, a new CPS is generated and connections with the S-cell at $\hat{\boldsymbol{n}}$ in the new CPS are reinforced. If some S-cells at $\hat{\boldsymbol{n}}$ in CPS's fire, the S-cell of maximum firing rate is selected and the connections with the S-cell are generated and reinforced. These processes can be explained by geometrical relation among a weight vector, an input vector and a threshold as shown in Figure 2. Let us consider a situation in which a pattern has already been trained in CPS#0. If an another input vector (training pattern) is in the accept-region of CPS#0, the vector is trained in CPS#0 (Figure 2(a)). On the other hand, if the input vector is outside the accept-region of CPS#0, a new CPS (CPS#1) is generated and the vector is trained in CPS#1 (Figure 2(b)).

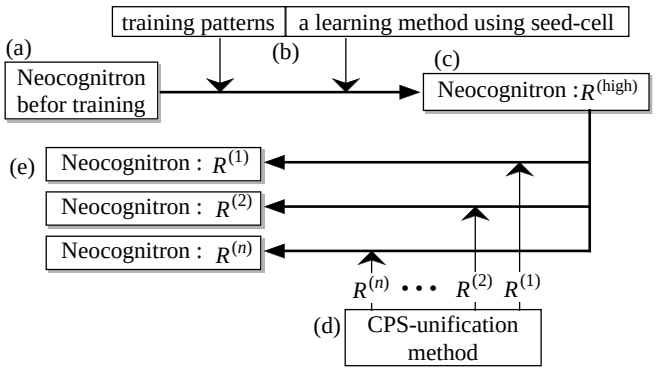

Figure 3: The flow diagram of auto-generating algorithm.

As a threshold of an S-cell becomes higher, more CPS's are generated, because accept-regions become narrower in a higher threshold.

We describe the "auto-generating algorithm" in the following. The flow diagram of the algorithm is shown in Figure 3. First, one constructs a neocognitron which is trained with a set of high thresholds, $R^{(\text{high})} = \left\{r_2^{(\text{high})}, r_3^{(\text{high})}, r_4^{(\text{high})}\right\}$, by presenting training patterns and applying a learning method using a seed-cell selecting plane (Figure 3(a)–(c)). We emphasize that training patterns are not presented any longer after the application of the learning method. An example in which two CPS's are generated with a narrow accept-region $\theta_l^{(\text{high})}$ is shown in Figure 4 (a).

The set of high thresholds, $R^{(\text{high})}$, is replaced by a set of lower threshold, $R^{(n)} = \left\{r_2^{(n)}, r_3^{(n)}, r_4^{(n)}\right\}$, in the process(d) in Figure 3. In the case of the lower thresholds the accept-region spreads to $\theta_l^{(n)}$ and the weight vector, $\boldsymbol{a}_l(\boldsymbol{n}, 1)$, in CPS#1 comes into the accept-region of CPS#0 as shown in Figure 4(b). This situation is inconsistent with the learning method using seed-cell selecting plane as mentioned above. This means that CPS#1 should not be generated and $\boldsymbol{a}_l(\boldsymbol{n}, 1)$ should be trained by an S-cell in CPS#0 with lower thresholds $R^{(n)}$. The CPS-unification method detects such an inconsistent situation between CPS#j and CPS#k which is described as

$$\cos^{-1}\frac{\boldsymbol{a}_l^{\mathrm{T}}(\boldsymbol{n}, j)\,\boldsymbol{a}_l(\boldsymbol{n}, k)}{||\boldsymbol{a}_l(\boldsymbol{n}, j)|| \cdot ||\boldsymbol{a}_l(\boldsymbol{n}, k)||} < \theta_l^{(n)}, \tag{15}$$

and unifies CPS#j and CPS#k into a newly generated CPS#i. The new connections with an S-cell of CPS#i are given by

$$a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, i) = a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, j) + a_l(\boldsymbol{\nu}, \boldsymbol{n}, \kappa, k), \tag{16}$$

$$b_l(i) = b_l(j) + b_l(k). \tag{17}$$

Such unification also influences the connections of S-cells

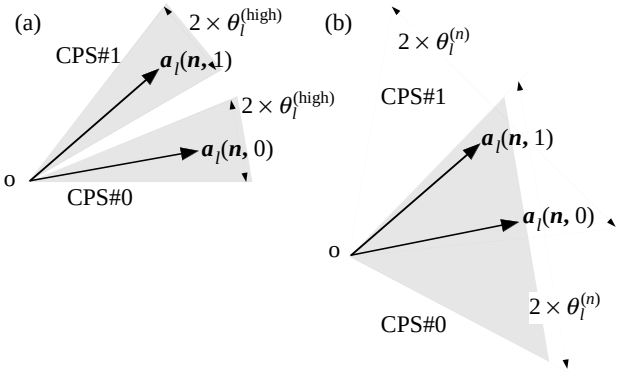

Figure 4: Schematic representation of a vector space with a different value of thresholds. (a): A high threshold. (b): A low threshold.

in the following module. The connections in the following module are given by

$$a_{l+1}(\boldsymbol{\nu}, \boldsymbol{n}, i, k') = a_{l+1}(\boldsymbol{\nu}, \boldsymbol{n}, j, k') + a_{l+1}(\boldsymbol{\nu}, \boldsymbol{n}, k, k'). \tag{18}$$

We note that in the new algorithm the training patterns are presented only once in order to construct many neocognitrons, while in a conventional algorithm the patterns are presented $|R_2| \times \cdots \times |R_L|$ times.

4. SIMULATION

We examine the ability of the rotation-invariant neocognitron for the recognition of realistic hand-written numerical patterns provided by ETL-1 database. We use five hundreds training patterns (see Figure 5 for examples) and one thousands test patterns. The rotation-invariant neocognitron can not distinguish between the pattern "6" and "9" in the last C-layer, $U_{\mathrm{C}4}$, because the model is completely robust even for the rotation of 180 degrees. Therefore, we observe a firing pattern of S-cells in $U_{\mathrm{S}4}$ preceding $U_{\mathrm{C}4}$. The parameters of the model are given in Table 1. The sizes of the areas, A_l and D_l, are given in Table 2.

Table 1: The number of cell-plane stack, and the number of cells in one cell-plane stack. The numbers in columns marked by asterisks are not defined, and the numbers in parentheses in columns are ones for the threshold, $R^{(\mathrm{opt})}$.

	$l=0$	$l=1$	$l=2$	$l=3$	$l=4$
$K_{\mathrm{S}l}$	*	1	(6)	(7)	(9)
$N_{\mathrm{S}l}$	*	59	17	13	3
$T_{\mathrm{S}l}$	*	16	8	4	2
$K_{\mathrm{C}l}$	1	1	(6)	(7)	(9)
$N_{\mathrm{C}l}$	61	19	17	10	1
$T_{\mathrm{C}l}$	1	8	4	2	1

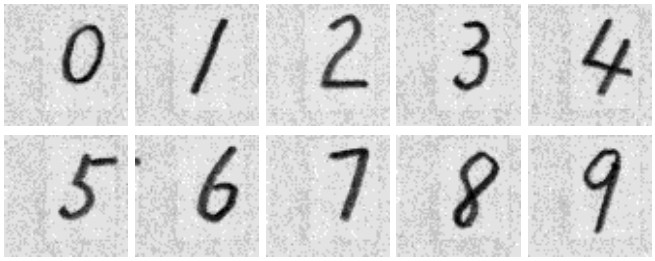

Figure 5: Examples of training patterns.

The number of the sets of thresholds is one thousand. We determine an optimal set of thresholds, $R^{(\mathrm{opt})}$, as the set that shows the best recognition rate for the training patterns using the auto-generating algorithm. The result turns out $R^{(\mathrm{opt})} = \{2.3, 2.0, 2.5\}$. The recognition rate of the rotation-invariant neocognitron with $R^{(\mathrm{opt})}$ shows 88.9% for the test patterns.

We examine the effectiveness and the efficiency of the auto-generating algorithm. We compare the rotation-invariant neocognitron based on the auto-generating algorithm with the model based on the conventional algorithm in the recognition rate and the number of CPS. In the conventional algorithm the recognition is 88.4%, almost same as the value that in the auto-generating algorithm. The numbers of CPS in both cases equal each other. It turns out that the model trained by auto-generating algorithm is almost equal to the model trained by the conventional algorithm. The auto-generating algorithm is efficient and effective in the sense that a presentation of training patterns and a learning is needed only once in order to construct many neocognitron. Since the execution time of the new algorithm is about eight seconds in generating one rotation-invariant neocognitron, the t_{test} becomes about eight thousands seconds. This means that we can reduce the time t_{test} from about two months to about two hours.

The rotation-invariant neocognitron is compared with a neocognitron in the recognition rate. It is very difficult for the neocognitron with ten C-cells in $U_{\mathrm{C}4}$ to learn correctly all five hundreds training patterns using

Table 2: A size of area connected with one cell. A_l and D_l denote the size connected with a C-cell and an S-cell, respectively.

	$l=1$	$l=2$	$l=3$	$l=4$
A_l	$3\times3\times1$	$3\times3\times8$	$5\times5\times4$	$3\times3\times2$
D_l	$5\times5\times3$	$3\times3\times3$	$5\times5\times3$	$3\times3\times2$

Figure 6: Examples of rotated patterns which are correctly recognized by a rotation-invariant neocognitron and can not recognized by an original neocognitron

an unsupervised learning, because the patterns of ETL-1 database are very distorted. For this reason we use twenty patterns to train both the models to make comparison on the same footing. In this case the neocognitron learns correctly all twenty patterns. The optimal set of thresholds of the neocognitron is $\{2.95, 2.48, 2.80\}$. For the one thousand test patterns the rotation-invariant neocognitron shows the recognition rate of 85.3%, and the original neocognitron shows the rate of 76.2%. Examples of rotated patterns shown in Figure 6 are correctly recognized by the rotation-invariant neocognitron but can not be recognized by the original neocognitron. We remark that the latest neocognitron [7] has additional layers, an edge-detecting layer or a bend-detecting layer [8], to improve the recognition rate, and shows the recognition rate of more than 98%.

Finally we examine the robustness for rotations. We make rotated patterns of $30°, 60°, \cdots$ and $330°$ for each one thousand test patterns. These amount to 22,000 in total number. The recognition rate for the rotated patterns is also 88.9%. The rotated patterns which are correctly recognized are made of the test patterns which are correctly recognized in the previous simulation. It turns out that our model is completely robust for rotations of realistic patterns. Examples of patterns which are not correctly recognized are shown in Figure 7. We expect that the rate can be improved by attaching a edge detecting layer or a bend-detecting layer [8], which has been introduced to an original neocognitron.

5. CONCLUSIONS

A new learning algorithm, auto-generating algorithm, using the new method, CPS-unification, is proposed, and the efficiency and the effectiveness of the method is shown. The recognition rate of the rotation-invariant neocognitron is higher than that of the original neocognitron. The rotation-invariant neocognitron is completely robust for rotations of realistic patterns. We intend to improve the recognition rate by adding an additional module used in an original neocognitron, and expect that the model will show a high recognition rate for realistic patterns.

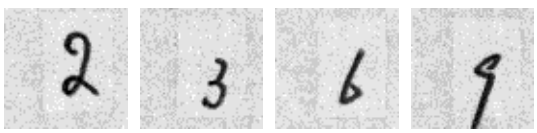

Figure 7: Examples of patterns recognized unsuccessfully.

Acknowledgments

Authors are grateful to Prof. K. Fukushima for useful discussions and suggestions. Authors wish to thank Prof. S. Inawashiro for discussions in preparing the final form of the paper.

References

[1] K. Fukushima : "Neocognitron: A hierarchical neural network capable of visual pattern recognition", *Neural Networks*, **Vol. 1**, pp. 119–130 (1988).

[2] S. Satoh, J. Kuroiwa, H. Aso and S. Miyake : "A rotation-invariant Neocognitron (in Japanese)", *IEICE Trans.* (to be published).

[3] S. Satoh, J. Kuroiwa, H. Aso and S. Miyake : "Recognition of rotated patterns using neocognitron(Proceeding) ", Proceedings of the International Conference on Neural Information Processing (ICONIP'97), **Vol. 1**, pp. 112–116 (1997).

[4] K. Fukushima and N. Wake : "An improved learning algorithm for the neocognitron(Proceeding)", Proceedings of the International Conference on Artificial Neural Networks ICANN'92, pp. 4–7 (1992).

[5] D.R. Lovell, T. Downs and A.C. Tsoi: "An Evaluation of The Neocognitron", *IEEE Trans.*, **Vol. 8**, pp. 1090–1105 (1997).

[6] K. Fukushima: "Analysis of the process of visual pattern recognition by the neocognitron", *Neural Networks*, **Vol. 2**, pp. 413–420 (1989).

[7] H. Shouno, K. Nagahara, K. Fukushima and M. Okada : "Neocognitron applied to hand-written Digit Recognition. —Evaluation with large Character Database— (in Japanese)", IEICE Tech. Rep., **NC97-19**, pp. 65–71 (1997).

[8] K. Fukushima and N. Wake: "Improved Neocognitron with Bend-Detecting Cells", International Joint Conference on Neural Networks IJCNN'92, **Vol. 4**, pp. 190-195 (1992).