AVAILABLE AT
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®

Neural Networks 17 (2004) 37–46

Neural Networks

www.elsevier.com/locate/neunet

Neocognitron capable of incremental learning

Kunihiko Fukushima*

School of Media Science, Tokyo University of Technology, 1404-1 Katakura Hachioji, Tokyo 192-0982, Japan

Received 21 October 2002; accepted 10 March 2003

Abstract

This paper proposes a new neocognitron that accepts incremental learning, without giving a severe damage to old memories or reducing learning speed. The new neocognitron uses a competitive learning, and the learning of all stages of the hierarchical network progresses simultaneously.

To increase the learning speed, conventional neocognitrons of recent versions sacrificed the ability of incremental learning, and used a technique of sequential construction of layers, by which the learning of a layer started after the learning of the preceding layers had completely finished. If the learning speed is simply set high for the conventional neocognitron, simultaneous construction of layers produces many garbage cells, which become always silent after having finished the learning. The proposed neocognitron with a new learning method can prevent the generation of such garbage cells even with a high learning speed, allowing incremental learning.

Keywords: Incremental learning; Neocognitron; Competitive learning; Multilayered neural network; Handwritten digit recognition

1. Introduction

The author previously proposed a neural network model *neocognitron* for robust visual pattern recognition (Fukushima, 1980; Fukushima & Miyake, 1982). It has a hierarchical multilayered architecture similar to the classical hypothesis of Hubel and Wiesel. It acquires the ability to recognize robustly visual patterns through learning.

To increase the learning speed, the neocognitrons of recent versions have sacrificed the ability of incremental learning. This paper proposes a neocognitron that is capable of incremental learning without reducing the learning speed. The new neocognitron has a modified network architecture and uses a new learning method. The new learning method allows the simultaneous construction of all stages of the network with a fast learning speed, and still accepts an incremental learning. The old memories made by an earlier learning will not be seriously destroyed by subsequent learning.

2. Conventional neocognitron

2.1. Simultaneous or sequential construction

The neocognitron consists of layers of S-cells, which resemble simple cells in the primary visual cortex, and layers of C-cells, which resemble complex cells. These layers of S-cells and C-cells are arranged alternately in a hierarchical manner.

S-cells are feature-extracting cells, whose input connections are variable and are modified through learning. C-cells, whose input connections are fixed and unmodified, exhibit an approximate invariance to the position of the stimuli presented within their receptive fields. The C-cells in the highest stage work as recognition cells, which indicate the result of the pattern recognition.

When a training stimulus is presented during the learning, each S-cell competes with other cells in its vicinity and has its input connections modified only when it wins the competition. The connections are modified so that the cell responds more strongly to the training stimulus to which the cell becomes a winner. If all S-cells in a competition area are silent, a new S-cell, and consequently a cell-plane is generated and learns the training stimulus.

* Tel.: +81-426-37-2469; fax: +81-426-37-2790.
E-mail address: fukushima@media.teu.ac.jp (K. Fukushima).

0893-6080/$ - see front matter © 2003 Elsevier Ltd. All rights reserved.
doi:10.1016/S0893-6080(03)00078-9

As for the sequence order of modifying connections of different layers, two alternative methods have been proposed. We will call these methods *simultaneous construction* and *sequential construction*. In the simultaneous construction, which is used in the original versions of the neocognitron (Fukushima, 1980; Fukushima & Miyake, 1982), learning of all layers in the network progresses simultaneously. In the sequential construction, which is used in most of the recent versions (Fukushima, 2003), learning starts from the lowest stage and progresses sequentially to higher stages: after the learning of a lower stage has been completely finished, the learning of the succeeding stage begins.

Both simultaneous and sequential constructions have merits and demerits for the conventional neocognitron. The simultaneous construction requires a slow learning speed but can accept incremental learning. On the other hand, the sequential construction can finish learning very fast, but does not accept incremental learning.

The proposed neocognitron allows the simultaneous construction of all stages of the network with a fast learning speed, and still accepts an incremental learning.

2.2. Conventional learning methods

We will first consider the case of simultaneous construction. Let U_{Sl} be the layer of S-cells in the lth stage of the network. The response of layer U_{Sl-1} of the preceding stage works as a training stimulus for layer U_{Sl}.[1] The number of cell-planes in U_{Sl-1} gradually increases with the progress of learning. An increase in the number of cell-planes in U_{Sl-1} means that the training stimulus given to U_{Sl} changes, even if the same training pattern is presented to the input layer U_0. If we express this in a multi-dimensional vector space, the dimension of learning vectors for U_{Sl} gradually increases with the progress of the learning of U_{Sl-1}.

If the learning speed of U_{Sl-1} is high, change in the response of U_{Sl-1}, which is caused by the increased number of cell-planes, occurs very fast. Because of the sudden change of signals from presynaptic cells, a cell of U_{Sl} often fails to respond to the training pattern, to which the cell used to become a seed cell. This situation is shown in Fig. 1(b). Since the cell cannot become a seed cell, the input connections to the cell-plane cannot be modified. The cell-plane fails to adapt to the fast change of layer U_{Sl-1} and stops responding for ever. Another cell-planes shall be generated now in U_{Sl} instead of the silent cell-plane. The silent cell-plane becomes garbage in the network and just consumes a large amount of computation time and memory when the network is installed in a computer.

[1] To be more strict, the response of C-cell layer U_{Cl-1}, which is a blurred version of the response of U_{Sl-1}, works as the training stimulus for U_{Sl}. To simplify the expression in this section, however, we write as though U_{Sl-1}, instead of U_{Cl-1}, is the training stimulus.

The generation of silent garbage cell-planes can be avoided, if the learning speed of the network is very slow. Because of a mechanism of shunting inhibition, the output of an S-cell is small when the connections to it are weak (See Eqs. (3), (5), (9) and (10), below). Hence the response of a cell-plane will stay small for a while after its generation, if the learning speed is slow. The response builds up gradually after that. In other words, the response of U_{Sl-1} to a training pattern, which becomes the training stimulus for U_{Sl}, does not make a rapid change even after the increase in the number of cell-planes in U_{Sl-1}. Therefore, each cell-plane of U_{Sl} can adapt to the increase in dimension of the training vector by shifting its reference vector gradually to the direction of the new training vector of an increased dimension. Although the generation of garbage cells can thus be avoided by a very slow learning speed, a large number of repeated presentations of the same leaning patterns are required before the learning finishes, because of slow building up of responses of the cells.

To increase the learning speed without generating garbage in the network, sequential construction is often used for the learning of the neocognitron of recent versions. Since the learning of U_{Sl} starts after the learning of U_{Sl-1} has completely finished, garbage cells will not be generated in U_{Sl} independent of the learning speed of U_{Sl-1}.

The sequential construction, however, does not accept incremental learning. Suppose an additional set of training patterns be supplied, after a network has already finished learning a certain set of training patterns. If a layer, which has learned the first training set, additionally learns the second training set, new cell-planes will usually be generated in the layer. Hence the layer comes to show different responses even to the patterns of the first training set. This is the same situation as in the case of simultaneous construction with a fast learning speed. Some of the cell-planes of the succeeding layer fail to respond even to the patterns to which they used to respond, and shall become garbage.

3. Basic idea of the learning

3.1. Response of an S-cell

To show the essence of the learning algorithm, we will extract only the circuit converging to a single S-cell (see Fig. 1(a) and analyze its behavior. Let a_i be the strength of the excitatory variable connection to an S-cell from the ith C-cell, whose output is x_i, and b be the inhibitory variable connection from the V-cell, whose output is v. Also let $c_i w_i$ be the strength of the excitatory connection to the inhibitory V-cell from the ith C-cell. The variable w_i, which will be discussed later in Section 3.2, is a kind of weight newly introduced in the proposed neocognitron, and is used to compensate the difference in the number of training among

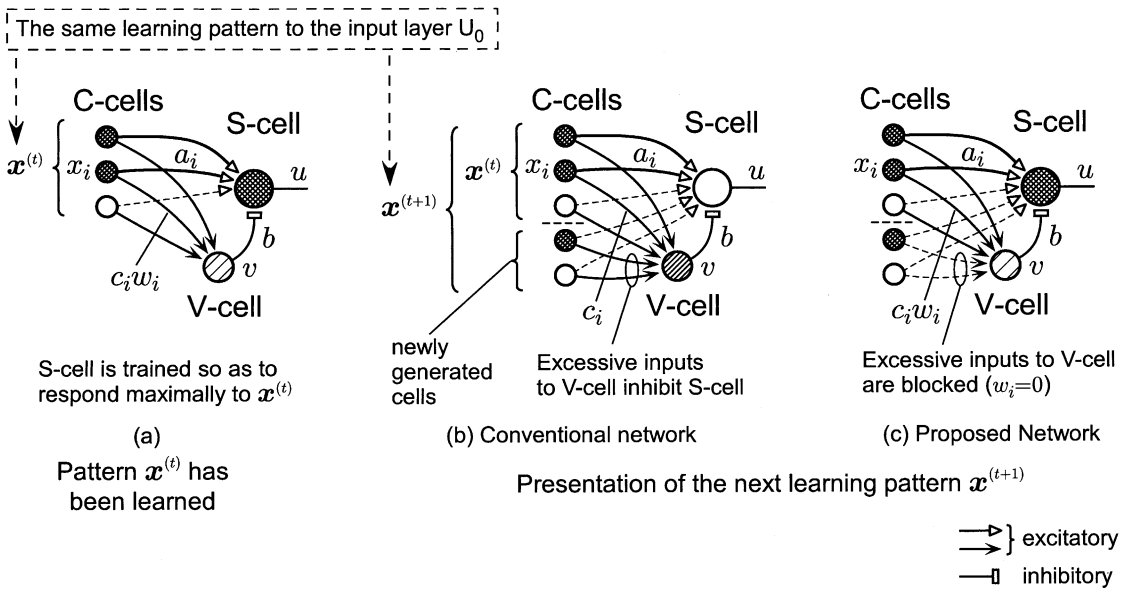

Fig. 1. Behavior of an S-cell during incremental learning. Comparison between the conventional and the proposed network.

a_i. Incidentally, we had $w_i = 1.0$ for the conventional neocognitron.

We also use vector notation **x** to represent the response of the presynaptic C-cells $\{x_i\}$. Similarly, vector **a** is used to represent connections $\{a_i\}$.

The output of the S-cell is given by

$$u = \frac{\theta}{1-\theta}\varphi\left[\frac{1+\sum_i a_i x_i}{1+\theta bv} - 1\right] \tag{1}$$

where $\varphi[\ ]$ is a function defined by $\varphi[x] = \max(x, 0)$. θ is a constant $(0 < \theta < 1)$ determining the threshold of the S-cell. The output of the V-cell is given by

$$v = \sqrt{\sum_i c_i w_i x_i^2} \tag{2}$$

If $bv \neq 0$ holds, Eq. (1) can also be written as follows:

$$u = \alpha\frac{\varphi[s-\theta]}{1-\theta} \tag{3}$$

where

$$s = \frac{\sum_i a_i x_i}{bv} \tag{4}$$

and

$$\alpha = \frac{\theta bv}{1+\theta bv} \tag{5}$$

Variable s represent a kind of similarity between the test vector **x** and the reference vector, which is expressed by **a**. Variable α can be considered as a constant (≈ 1) after some progress of learning, in which the inhibitory connection b has become large enough to satisfy $\theta bv \gg 1$. If the input to the S-cell is completely zero, however, we have $v = 0$ and $\alpha = 0$.

During the learning, each S-cell competes with other cells in its vicinity, and the winners of the competition become seed cells and learn the training pattern. Although a large number of training patterns are presented to the neocognitron, only a portion of them makes this particular S-cell a winner (or a seed cell). The vector **x** that makes this S-cell a winner becomes a training vector for this S-cell, and the tth training vector for this S-cell is represented by $\mathbf{x}^{(t)}$. To simplify the discussion, we assume here, without loosing generality, that the same S-cell is always selected as the seed cell from the cell-plane.

In an intermediate stage of the network, the number of presynaptic C-cells, namely, the dimension of the training vector $\mathbf{x}^{(t)}$, gradually increases with the progress of learning, because of simultaneous construction of all stages of the hierarchical network. A training vector presented earlier usually has a smaller dimension.

Every time when the S-cell becomes a winner, the excitatory connection a_i is strengthened by an amount proportional to the response of the presynaptic C-cells. If the C-cell has not been generated yet, a_i does not change. Namely

$$\Delta a_i = \begin{cases} q c_i x_i^{(t)} & \text{if } t \geq T_i \\ 0 & \text{if } t < T_i \end{cases} \tag{6}$$

where T_i is the time when the ith presynaptic C-cell is generated. Parameter q is a positive constant determining the learning speed. Since $a_i \approx 0$ in the initial state, a_i after

finishing T times of learning is

$$a_i = qc_i \sum_{t=T_i}^{T} x_i^{(t)} \tag{7}$$

This means that vector components of missing dimensions are treated as zero, when a training vector of a smaller dimension is summed up to **a**.

The inhibitory connection b is determined directly from the values of the excitatory connections a_i and weight w_i. That is

$$b = \sqrt{\sum_i \frac{a_i^2}{c_i w_i}} \tag{8}$$

3.2. Hypothetical case

Since the learning of all layers of the network progresses simultaneously, the response of a presynaptic C-cell is not always the same during the learning, even though the same training pattern is presented to the input layer of the network. The weight w_i is used to compensate this effect.

We will now consider a very simple hypothetical case as a first approximation, and discuss how to determine weight w_i.

We hypothesize as follows. The S-cell be selected as a seed only when the same training pattern is presented to the input layer U_0 of the network. Let $x_i^{(t)}$ be the response of the ith presynaptic C-cell when this same training pattern is presented at the tth time. Let ξ_i be the response of this C-cell after finishing enough times of learning of the preceding stages of the network. We also use vector notation $\boldsymbol{\xi}$ to represent the vector whose ith component is ξ_i.

In an early period of the learning, the response $x_i^{(t)}$ is reduced from ξ_i by a factor of α_τ, which is a kind of gain of the C-cell, where τ represents the elapsed time since the cell has been generated. That is

$$x_i^{(T_i+\tau-1)} = \begin{cases} \alpha_\tau \xi_i & \text{if } \tau \geq 1 \\ 0 & \text{if } \tau \leq 0 \end{cases} \tag{9}$$

Gain α_τ increases asymptotically to 1.0 with the progress of the learning of the preceding stages. We assume here that

$$\alpha_\tau = \frac{\tau}{\tau + \sigma} \tag{10}$$

Eqs. (9) and (10) are obtained approximately under an assumption that the responses of presynaptic C-cells, which relay the outputs of S-cells, also follow equations like (3) and (5). In other words, ξ_i corresponds to $\varphi[s - \theta]/(1 - \theta)$, and α_τ to α, in Eq. (3).

If the above hypothesis holds, we have from Eqs. (7), (9) and (10)

$$a_i = qc_i\xi_i \sum_{\tau=1}^{T-T_i+1} \alpha_\tau = qc_i\xi_i W(m_i) \tag{11}$$

where $W(m)$ is a function defined by

$$W(m) = \sum_{\mu=1}^{m} \alpha_\mu = \sum_{\mu=1}^{m} \frac{\mu}{\mu + \sigma} \tag{12}$$

and m_i represents how many times the output of the ith presynaptic C-cell has been used for the training of this S-cell. Namely

$$m_i = T - T_i + 1 \tag{13}$$

This situation is shown in Fig. 2. In this example, the first and the second presynaptic C-cells have been generated since the beginning of the learning, and their outputs are summed up three times to **a**. Since the third C-cell has not been generated yet at $t = 1$, its response is treated as zero when the vectors $\mathbf{x}^{(1)}$ is summed up to **a**. The fourth C-cell, which is generated only at $t = 3$, is summed only once. The test vector **x**, which will be presented later, might have more components corresponding to C-cells generated afterward.

Substituting Eqs. (2), (8) and (11) in Eq. (4), we have s for an arbitrary test vector **x**.

$$s = \frac{\sum_i c_i W(m_i)\xi_i x_i}{\sqrt{\sum_i c_i \dfrac{\{W(m_i)\xi_i\}^2}{w_i}} \sqrt{\sum_i c_i w_i x_i^2}} \tag{14}$$

If we adjust the value of weight w_i to

$$w_i = W(m_i) \tag{15}$$

Eq. (14) reduces to

$$s = \frac{\sum_i c_i w_i \xi_i x_i}{\sqrt{\sum_i c_i w_i \xi_i^2} \sqrt{\sum_i c_i w_i x_i^2}} \tag{16}$$

Eq. (15) means that a heavier weight w_i is assigned to the signal from a C-cell that have been used more frequently for the training.

$$\begin{array}{lccccc} & c_1 & c_2 & c_3 & c_4 & \\ \boldsymbol{x}^{(1)} = (& \alpha_1\xi_1, & \alpha_1\xi_2 &) & & \\ \boldsymbol{x}^{(2)} = (& \alpha_2\xi_1, & \alpha_2\xi_2, & \alpha_1\xi_3 &) & \\ \boldsymbol{x}^{(3)} = (& \alpha_3\xi_1, & \alpha_3\xi_2, & \alpha_2\xi_3, & \alpha_1\xi_4 &) \\ \hline \boldsymbol{a} = q(& c_1W(3)\xi_1, & c_2W(3)\xi_2, & c_3W(2)\xi_3, & c_4W(1)\xi_4 &) \\ \boldsymbol{x} = (& x_1, & x_2, & x_3, & x_4, & x_5) \end{array} \qquad W(m) = \sum_{\mu=1}^{m} \alpha_\mu$$

Fig. 2. Relation between connection **a** and the test vector **x**.

Here we define a *weighted* inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ by $(\mathbf{x},\mathbf{y}) = \sum_i c_i w_i x_i y_i$, where the strength of the connections converging to the inhibitory V-cell, $c_i w_i$, is used as the weight. We also define the norm of a vector $\mathbf{x}$ by $\|\mathbf{x}\| = \sqrt{(\mathbf{x},\mathbf{x})}$. Using this weighted inner product, Eq. (16) can be represented by

$$s = \frac{(\boldsymbol{\xi},\mathbf{x})}{\|\boldsymbol{\xi}\|\cdot\|\mathbf{x}\|} \tag{17}$$

We can interpret that s represents a kind of similarity between the training vector $\boldsymbol{\xi}$ and the test vector $\mathbf{x}$. It takes the maximum value of 1.0 when $\mathbf{x} = \boldsymbol{\xi}$. As can be seen from Eq. (3), the S-cell yields a non-zero output for $s > \theta$. In the vector space of $\mathbf{x}$, the conical area determined by $s > \theta$ around the training vector $\boldsymbol{\xi}$ becomes the *tolerance area* of the S-cell. The S-cell responds if and only if the test vector $\mathbf{x}$ falls in the tolerance area.

4. Network architecture

Based on the discussions in the previous section, we propose a new neocognitron. As shown in Fig. 3, the proposed neocognitron has almost the same network architecture as the neocognitron of a recent version, which is designed for handwritten digit recognition (Fukushima, 2003). Only the difference between them resides in V-cell-planes (namely, cell-planes of V-cells). In the conventional neocognitron, each layer of S-cells has only one V-cell-plane, which is used in common for all S-cell-planes of the layer. In the proposed neocognitron, each S-cell-plane has its own V-cell-plane. In other words, each S-cell has its own V-cell.

The stimulus pattern is presented to the input layer U_0. A layer of contrast-extracting cells (U_G), which correspond to retinal ganglion cells or lateral geniculate nucleus cells, follows layer U_0. Layer U_G consists of two cell-planes: a cell-plane consisting of cells with concentric on-center receptive fields, and a cell-plane consisting of cells with off-center receptive fields. The former cells extract positive contrast in brightness, whereas the latter extract negative contrast from the images presented to the input layer.

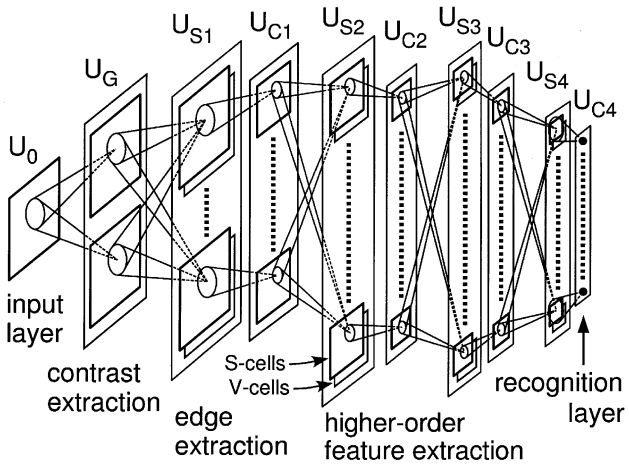

Fig. 3. The architecture of the proposed neocognitron.

The output of layer U_G is sent to the S-cell layer of the first stage (U_{S1}). The S-cells of layer U_{S1} correspond to simple cells in the primary visual cortex. They have been trained using supervised learning to extract oriented edge components from the input image (Fukushima, 2003).

The present model has four stages of S- and C-cell layers. The output of layer U_{Sl} (S-cell layer of the lth stage) is fed to layer U_{Cl}, where a blurred version of the response of layer U_{Sl} is generated. An inhibitory surround is introduced around the excitatory connections of the input connections to each C-cell (Fukushima, 2003). The density of the cells in each cell-plane is reduced between layers U_{Sl} and U_{Cl}. Layer U_{C4}, which is in the highest stage of the network, is the recognition layer, whose response shows the final result of pattern recognition by the network.

The S-cells of layers U_{S2}, U_{S3} and U_{S4} are self-organized using the proposed method.

Since main difference from the conventional neocognitron (Fukushima, 2003) resides in S-layers, we will show mathematical expressions of the response of a layer of S-cells only.

Let $u_{Sl}(\mathbf{n},k)$, $v_l(\mathbf{n},k)$ and $u_{Cl}(\mathbf{n},k)$ be the output of S-, V- and C-cells of the kth cell-plane of the lth stage, respectively, where $\mathbf{n}$ represents the location of the receptive field center of the cells, and k is the sequence number of the cell-plane. The outputs of S-cells are given by

$$u_{Sl}(\mathbf{n},k) = \frac{\theta_l}{1-\theta_l}\cdot\varphi\left[\frac{1+\sum_{\kappa=1}^{K_{Cl-1}}\sum_{|\boldsymbol{\nu}|<A_{Sl}} a_{Sl}(\boldsymbol{\nu},\kappa,k)\cdot u_{Cl-1}(\mathbf{n}+\boldsymbol{\nu},\kappa)}{1+\theta_l\cdot b_{Sl}(k)\cdot v_l(\mathbf{n},k)} - 1\right] \tag{18}$$

where $\varphi[\,]$ is a function defined by $\varphi[x]=\max(x,0)$. Parameter $a_{Sl}(\boldsymbol{\nu},\kappa,k)$ (≥ 0) is the strength of variable excitatory connection coming from C-cell $u_{Cl-1}(\mathbf{n}+\boldsymbol{\nu},\kappa)$ of the preceding stage. For $l=1$, however, $u_{Cl-1}(\mathbf{n},k)$ stands for $u_G(\mathbf{n},k)$, and we have $K_{Cl-1}=2$. It should be noted here that all cells in a cell-plane share the same set of input connections, hence $a_{sl}(\boldsymbol{\nu},\kappa,k)$ is independent of $\mathbf{n}$. A_{Sl} denotes the radius of summation range of $\boldsymbol{\nu}$, that is, the size of spatial spread of input connections to a particular S-cell. Parameter $b_l(k)$ (≥ 0) is the strength of variable inhibitory connection coming from the V-cell. The positive constant θ_l is the threshold of the S-cell and determines the selectivity in extracting features. Incidentally, if we replace $v_l(\mathbf{n},k)$ with $v_l(\mathbf{n})$, Eq. (18) becomes the same as that for the conventional neocognitron.

On the other hand, the outputs of V-cells are given by

$$v_l(\mathbf{n},k)=\sqrt{\sum_{\kappa=1}^{K_{Cl-1}}\sum_{|\boldsymbol{\nu}|<A_{Sl}} c_{Sl}(\boldsymbol{\nu})\cdot w_l(\kappa,k)\cdot\{u_{Cl-1}(\mathbf{n}+\boldsymbol{\nu},\kappa)\}^2} \quad (19)$$

The excitatory input connection from the κth C-cell-plane consists of two components: a variable weight $w_l(\kappa,k)$, and a fixed constant $c_{Sl}(\boldsymbol{\nu})$, which is a monotonically decreasing function of $|\boldsymbol{\nu}|$.

In an intermediate stage of the network, K_{Cl-1}, the number of presynaptic C-cell-planes, does not stay constant, but increases, during the learning, because simultaneous construction progresses for all stages of the hierarchical network. As was discussed in Section 3, excitatory connections $a_{Sl}(\boldsymbol{\nu},\kappa,k)$ from layer U_{Cl-1} are increased by an amount proportional to the response of C-cells presynaptic to the seed cell, and a cell-plane that was created earlier (say, the cell-plane with $\kappa=1$) has a larger contribution to strengthening the connections than a newly created cell-plane, because it has been presented more frequently to the seed cell. The difference in contribution to the excitatory connections is reflected to the connections to the inhibitory V-cell through weight $w_l(\kappa,k)$, which is also modified by the learning.

5. Learning method

We adopt simultaneous construction, by which learning of all layers progresses simultaneously in the network.

5.1. Intermediate stages

The S-cells of intermediate stages (U_{S2} and U_{S3}) are self-organized using unsupervised competitive learning similar to the method used in the conventional neocognitron (Fukushima, 2003).

During the learning, each S-cell competes with other cells in its vicinity, and the winners of the competition become seed cells. Once the seed cells are determined, variable connections $a_{sl}(\boldsymbol{\nu},\kappa,k)$ and $b_{sl}(k)$ are strengthened depending on the responses of the C-cells presynaptic to the seed cells.

Let cell $u_{Sl}(\hat{\mathbf{n}},\hat{k})$ be selected as a seed cell at a certain time. Variable connections $a_{sl}(\boldsymbol{\nu},\kappa,\hat{k})$ to this seed cell, and consequently to all S-cells in the same cell-plane as the seed cell, are increased by the following amount:

$$\Delta a_{Sl}(\boldsymbol{\nu},\kappa,\hat{k}) = \begin{cases} q_l\cdot c_{Sl}(\boldsymbol{\nu})\cdot u_{Cl-1}(\hat{\mathbf{n}}+\boldsymbol{\nu},\kappa) & \text{if } 1\le\kappa\le K_{Cl-1} \\ 0 & \text{if } \kappa>K_{Cl-1} \end{cases} \quad (20)$$

It should be noted here that the number of the presynaptic C-cell-planes, K_{Cl-1}, is not constant but increases with the progress of learning. Eq. (20) means that $a_{Sl}(\boldsymbol{\nu},\kappa,\hat{k})$ does not change if the presynaptic cell $u_{Cl-1}(\hat{\mathbf{n}}+\boldsymbol{\nu},\kappa)$ has not been generated yet. Positive constant q_l determines the learning speed.

Weight $w_l(\kappa,\hat{k})$ is determined by the following equations

$$w_l(\kappa,\hat{k}) = W(m_l(\kappa,\hat{k})) \quad (21)$$

where $W(m)$ is a function defined by

$$W(m)=\sum_{\mu=1}^{m}\frac{\mu}{\mu+\sigma} \quad (22)$$

and $m_l(\kappa,\hat{k})$ represents how many times the κth C-cell-plane has been used for the training of the $\hat{k}$th S-cell-plane. Namely, $m_l(\kappa,\hat{k})$ is increased by one every time when the S-cell is selected as a seed, if the κth C-cell-plane has already been generated at that moment:

$$\Delta m_l(\kappa,\hat{k}) = \begin{cases} 1 & \text{if } 1\le\kappa\le K_{Cl-1} \\ 0 & \text{if } \kappa>K_{Cl-1} \end{cases} \quad (23)$$

The inhibitory connection $b_{sl}(\hat{k})$ is determined directly from the values of the excitatory connections $a_{sl}(\boldsymbol{\nu},\kappa,\hat{k})$ and the corresponding weights $w_l(\kappa,\hat{k})$:

$$b_{Sl}(\hat{k})=\sqrt{\sum_{\kappa=1}^{K_{Cl-1}}\sum_{|\boldsymbol{\nu}|<A_{Sl}}\frac{\{a_{Sl}(\boldsymbol{\nu},\kappa,\hat{k})\}^2}{c_{Sl}(\boldsymbol{\nu})\cdot w_l(\kappa,k)}} \quad (24)$$

For other cell-planes from which no seed cell is selected ($k\neq\hat{k}$), all of these values, namely, $a_{Sl}(\boldsymbol{\nu},\kappa,k)$, $w_l(\kappa,k)$ and $b_{Sl}(k)$, do not changed at this moment.

The method of dual threshold (Fukushima & Tanigawa, 1996) is also used for the learning: Competition among S-cells is based on the responses with a high threshold value θ_l^{L}, and signals that are sent to the succeeding stage are calculated with a lower threshold value θ_l^{R}.

To be more specific, outputs of S-cells are calculated by equation Eq. (18) using threshold value of $\theta_l=\theta_l^{\mathrm{L}}$. Competition among S-cells is based on these outputs. Once seed cells are selected from U_{Sl} and the connections have been strengthened by Eqs. (20), (23) and (24), the outputs of S-cells are calculated again using the updated connections and the lower threshold θ_l^{R}. The latter outputs of S-cells are then sent to the succeeding stage. Incidentally, during the recognition phase after having finished learning, the lower threshold θ_l^{R} is always used.

5.2. The highest stage

S-cells of the highest stage (U_{S4}) are trained using a supervised competitive learning (Fukushima, 2003). The learning rule resembles the competitive learning used for training U_{S2} and U_{S3}, but the class names of the training

patterns are also utilized for the learning. When the network learns varieties of deformed training patterns through competitive learning, more than one cell-plane for one class is usually generated in U_{S4}. Therefore, when each cell-plane first learns a training pattern, the class name of the training pattern is assigned to the cell-plane. Thus, each cell-plane of U_{S4} has a label indicating one of the 10 digits.

Every time a training pattern is presented, competition occurs among all S-cells in the layer. If the winner of the competition has the same label as the training pattern, the winner becomes the seed cell and learns the training pattern in the same way as the seed cells of the lower stages. If the winner has a wrong label (or if all S-cells are silent), however, a new cell-plane is generated and is put a label of the class name of the training pattern.

Competition among S-cells occurs also in the recognition phase, and the label of the maximum-output S-cell of U_{S4} determines the final result of recognition. We can also express this process of recognition as follows. Recognition layer U_{C4} has 10 C-cells corresponding to the 10 digits to be recognized. Every time when a new cell-plane is generated in layer U_{S4} during the learning, excitatory connections are created from all S-cells of the cell-plane to the C-cell of that class name. Only one maximum output S-cell within the whole layer U_{S4} can transmit its output to U_{C4}.

In the conventional neocognitron, the threshold of the highest stage for the learning phase was chosen as low as that for the recognition phase, namely, $\theta_4^{\mathrm{L}} = \theta_4^{\mathrm{R}}$ (Fukushima, 2003). In the present system, however, a higher threshold value is used for the learning phase because of the following reason. A low threshold produces a large tolerance area around each reference vector. If we use a low threshold, the border between two arbitrary classes is adjusted step by step by generating new cell-planes for erroneously recognized training patterns, and finally stabilizes by a power balance of the two classes. In experiment B, which will be discussed in Section 6.2 below, for example, any of the training patterns for classes '0'–'4' are never presented during the second phase of the learning, in which the network learns training patterns for classes '5'–'9'. Since no training pattern from a class is presented in the second phase of the learning, its class border might be invaded by other classes. If the threshold is high, however, each cell-plane has a small tolerance area around its reference vector, and does not invade deep into the territories of other classes. We chose this high threshold to protect old memories from destruction by incremental learning.

The threshold, however, should not be so high as $\theta_4^{\mathrm{L}} \approx 1$. If we have $\theta_4^{\mathrm{L}} \approx 1$, each training pattern generates its own cell-plane, and layer U_{S4} behaves like a nearest-neighbor classifier.

6. Computer simulation

6.1. Scale of the network

We tested the behavior of the proposed network by computer simulation using handwritten digits (free writing) randomly sampled from the ETL1 database.[2] Incidentally, the ETL1 is a database of segmented handwritten characters. The network has the same scale and parameters as the one reported by Fukushima (2003), except the number of V-cell-planes. That is, the total number of cells (not counting inhibitory V-cells) in each layer is: $U_0 : 65 \times 65$, $U_G : 71 \times 71 \times 2$, $U_{S1} : 68 \times 68 \times 16$, $U_{C1} : 37 \times 37 \times 16$, $U_{S2} : 38 \times 38 \times K_{S2}$, $U_{C2} : 21 \times 21 \times K_{C2}$, $U_{S3} : 22 \times 22 \times K_{S3}$, $U_{C3} : 13 \times 13 \times K_{C3}$, $U_{S4} : 5 \times 5 \times K_{S4}$, $U_{C4} : 1 \times 1 \times 10$. Although the number of cells in each cell-plane has been pre-determined for all layers, the number of cell-planes in an S-cell layer (K_{Sl}) is determined automatically during the learning depending on the training set. In each stage except the highest one, the number of cell-planes of the C-cell layer (K_{Cl}) is the same as K_{Sl}. The recognition layer U_{C4} has $K_{C4} = 10$ cell-planes corresponding to ten digits, and each cell-plane contains only one C-cell. The thresholds of S-cells were chosen as follows. For the edge-extracting layer U_{S1}, we chose $\theta_1 = 0.55$. For the higher layers U_{S2}, U_{S3} and U_{S4}, the thresholds for the recognition (namely, thresholds for calculating responses of S-cells) were $\theta_2^{\mathrm{R}} = 0.51$, $\theta_3^{\mathrm{R}} = 0.58$ and $\theta_4^{\mathrm{R}} = 0.30$. Those for the learning (namely, thresholds used for the competition) were: $\theta_2^{\mathrm{L}} = 0.66$, $\theta_3^{\mathrm{L}} = 0.67$. As for the highest stage, however, we used $\theta_4^{\mathrm{L}} = 0.72$, instead of $\theta_4^{\mathrm{L}} = 0.30$ that was used for the previous network (Fukushima, 2003). Parameter σ in Eq. (22) (or Eq. (12) was adjusted to 0.5.

6.2. Recognition rate

To demonstrate that the network accepts incremental learning, we will show how the recognition rate changes depending on two different ways of pattern presentation in the learning.

Experiment A. The network learns a single training set consisting of patterns of all classes, namely, handwritten digits from '0' to '9'. The training set has the same number of patterns from each class. The patterns are randomly sampled from the ETL1 database. To test how the recognition rate changes depending on the total number of patterns in the training set, we prepared training sets consisting of 500, 1000, 2000 and 3000 patterns. The training set of 500 patterns is a subset of 1000, which in tern is a subset of 2000, and so on. The patterns in a training set are presented in the order of '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '1',… The training set is presented three times to the network.

[2] ETL1 database: http://www.etl.go.jp/~etlcdb/index.htm.

Table 1
Recognition rates of the network for experiments A and B using training sets of different sizes

Network	Method of constructing layers	Training patterns		Recognition rate (%) for test set (for training set)				Scale of the network		
		Method of presentation	Number of patterns	After	'0'–'4'	'5'–'9'	'0'–'9'	K_{S2}	K_{S3}	K_{S4}
New	Simultaneous	Together (experiment A)	500		96.2 (100)	97.0 (100)	**96.6** (100)	28	86	74
			1000		96.5 (99.8)	98.2 (100)	**97.3** (99.9)	36	104	103
			2000		98.2 (99.9)	98.5 (100)	**98.3** (100)	43	128	146
			3000		98.1 (100)	98.5 (99.9)	**98.3** (100)	45	140	179
		Incremental (experiment B)	500	1/2	98.3 (100)	–	–	24	64	27
				2/2	94.1 (97.6)	96.5 (100)	**95.3** (98.8)	28	78	58
			1000	1/2	98.4 (100)	–	–	26	72	33
				2/2	95.0 (99.0)	97.5 (100)	**96.3** (99.5)	32	85	76
			2000	1/2	99.1 (100)	–	–	32	82	46
				2/2	97.5 (99.3)	98.6 (100)	**98.0** (99.7)	42	108	110
			3000	1/2	99.3 (100)	–	–	35	91	57
				2/2	98.1 (99.1)	98.7 (99.9)	**98.4** (99.5)	48	129	126
Old	Sequential	Together (experiment A)	3000		98.3 (100)	98.9 (100)	**98.6** (100)	39	110	103
	Simultaneous		3000		98.4 (100)	98.2 (99.7)	**98.3** (99.9)	45	241	325

After having finished the learning, the recognition rate of the network is measured using a blind test set. The blind test set consists of 3000 digits (300 patterns from each class), which are also randomly sampled from the ETL1 database, but there is no overlapping of patterns between the training and the test sets.

Experiment B. The training set used for experiment A is divided into two: one containing digits from '0' to '4', and the other from '5' to '9'. The network initially learns the first training set. The patterns are presented in the order of '0', '1', '2', '3', '4', '0', '1', '2',… After having finished learning the first training set, the network learns the second training set. After having finished the first and the second half of the learning, the recognition rate of the network was measured using the blind test set. It should be noted here that no more additional learning for the first training set is made after starting the second learning.

Table 1 summarizes the recognition rates under these various conditions. The numbers in the parentheses are the recognition rates for the training sets. The recognition rates

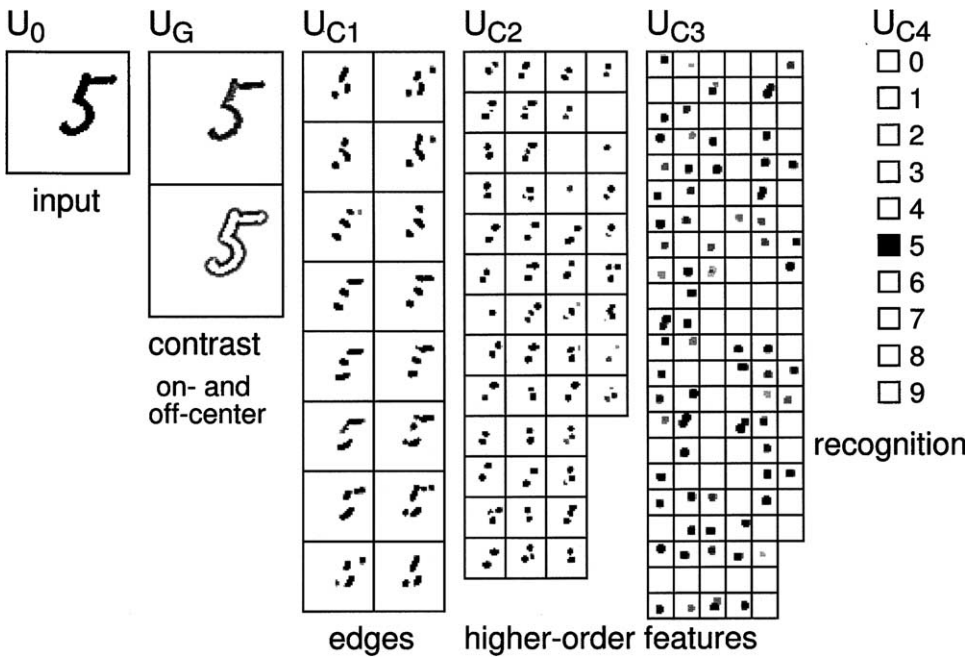

Fig. 4. An example of the response of the neocognitron. It has learned a training set consisting of '0'–'4' first, then a set of '5'–'9'. The input pattern is recognized correctly as '5'.

for '0'–'4' and '5'–'9' have been separately counted, as well as for all '0'–'9' patterns, in the table. The numbers of cell-planes generated when the learning has finished are also listed. As for experiment B, recognition rates when the first phase of the learning has just finished are also listed in the table.

As can be seen from the table, the recognition rates produced by experiments A and B are almost the same. They are sometimes slightly better for experiment A, and sometimes better for experiment B, depending on the number of training patterns. When we used 3000 training patterns, for example, recognition rates for experiment A and B were 98.3 and 98.4%, respectively. It is interesting to note that the numbers of cell-planes are smaller for experiment B. These results show that the memory of the first learning is not seriously destroyed by the second learning. In other words, the proposed neocognitron accepts incremental learning without giving a serious damage to old memories.

For your information, the second line from the bottom of Table 1 shows the results for the conventional neocognitron, in which the learning progresses sequentially from lower to higher stages (Fukushima, 2003). Comparing with these results, we can see that the new neocognitron produces a comparably high recognition rate without increasing the numbers of cell-planes so much, even if simultaneous construction is adopted, or even if incremental learning is used.

Incidentally, if simultaneous construction was applied to the conventional network with the same parameters, extremely large number of cell-planes were generated as shown at the bottom line of Table 1. Such a large number of cell-planes would not be practically acceptable. Probably many garbage cells were generated in the network. The recognition rate, however, was 98.3% and was almost the same as that for the proposed network.

Fig. 4 shows a response of the network that has finished the learning by experiment B using the training set of 3000 patterns. The responses of layers U_0, U_G, U_{C1}, U_{C2}, U_{C3} and U_{C4} are displayed in series from left to right. The rightmost layer, U_{C4}, is the recognition layer, whose response shows the final result of recognition.

7. Discussion

This paper has proposed a new neocognitron that accepts incremental learning, without giving a severe damage to old memories or reducing learning speed. The new neocognitron uses a competitive learning, and the learning of all stages of the hierarchical network progresses simultaneously with a fast learning speed. Cells in higher stages adapt to the change of the preceding stages of the network, and the generation of garbage cells, which are always silent after having finished the learning, can be prevented.

Various systems for incremental learning have been proposed so far (Carpenter, Grossberg, & Reynolds, 1995; Hoya & Chambers, 2001; Molina & Niranjan, 1996; Platt, 1991; Polikar, Udpa, Udpa, & Honavar, 2001; Williamson, 1996; Yamauchi, 2001; Yamauchi, Yamaguchi, & Ishii, 1999; Yingwei, Sundararajan, & Saratchandran, 1997). Most of them combines various methods to protect old memories from destruction by the incremental learning or restore the damaged memories.

Some of them try to remove garbage cells that have been generated during the learning. Another group of methods try to retrain the garbage cells for other purposes. It is difficult, however, to judge if a cell is garbage or not, once the cell has been generated in the network. We cannot use firing probability or response strength of the cells as a criterion for finding out garbage cells. Under experiment B discussed in Section 6.2, for example, any of the training patterns for classes '0'–'4' are never presented during the second phase of the learning, in which the network learns training patterns for classes '5'–'9'. Since training patterns for classes '0'–'4' are never presented during the second phase of the learning, the cells that are indispensable for the recognition of '0'–'4' might not respond at all. We should not remove cells simply because they are silent during the second phase of the learning.

Some methods use decay or forgetting factor in the variable connections to adapt to the changing probability of occurrence of the training patterns. These methods, however, might not work well for a situation like experiment B. Memories for '0'–'4' might fade out while learning '5'–'9'.

In contrast to these conventional learning methods, our learning method suppresses the generation of garbage cells, rather than removing them after having been generated. Cells in higher stages adapt to the change of the preceding stages of the hierarchical network, so as to continue executing their mission. Since cells can change their input connections only when they become winners, old memories stored in the connections are not destroyed even if the cells are silent for a long time.

Some other conventional methods propose to store typical examples of old training patterns and relearn them to restore old memories damaged by the incremental learning. Our learning method does not require such storage, either.

In Section 3.2, we assumed that α_τ can be approximated by Eq. (10) and put $W(m) = \sum_{\mu=1}^{m} \mu/(\mu+\sigma)$. Thinking of a possibility of simplifying the learning rule, we tried another approximation for α_τ, that is, $\alpha_\tau = 1$, by which we can have $W(m) = m$. Computer simulation showed, however, that the recognition rate under this simpler approximation was a little worse than that for Eq. (10).

Acknowledgements

The author thanks Katsuyoshi Yanagawa, graduate student at the University of Electro-Communications (now

at Hitachi Software Engineering Co.), for his great contribution to this work. He first proposed the use a separate V-cell-plane for each S-cell-plane (Yanagawa, Fukushima, & Yoshida, 2002). This work was partially supported by Grant-in-Aid-for-Scientific-Research #14380169, and Special Coordination Fund for Promoting Science and Technology (Project on Neuroinformatics Research in Vision), both from the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

Carpenter, G. A., Grossberg, S., & Reynolds, J. H. (1995). A fuzzy ARTMAP nonparametric probability estimator for nonstationary pattern recognition problems. *IEEE Transactions on Neural Networks*, *6*(6), 1330–1336.

Fukushima, K. (1980). Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, *36*(4), 193–202.

Fukushima, K. (2003). Neocognitron for handwritten digit recognition1. *Neurocomputing*, *51C*, 161–180.

Fukushima, K., & Miyake, S. (1982). Neocognitron: a new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, *15*(6), 455–469.

Fukushima, K., & Tanigawa, M. (1996). Use of different thresholds in learning and recognition. *Neurocomputing*, *11*(1), 1–17.

Hoya, T., & Chambers, J. A. (2001). Heuristic pattern correction scheme using adaptively trained generalized regression neural networks. *IEEE Transactions on Neural Networks*, *12*(1), 91–100.

Molina, C., & Niranjan, M. (1996). Pruning with replacement on limited resource allocating networks by F-projections. *Neural Computation*, *8*(4), 855–868.

Platt, J. (1991). A resource allocating network for function interpolation. *Neural Computation*, *3*(2), 213–225.

Polikar, R., Udpa, L., Udpa, S. S., & Honavar, V. (2001). Learn ++ an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man and Cybernetics, Part C*, *31*(4), 497–508.

Williamson, J. R. (1996). Gaussian ARTMAP: a neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, *9*(5), 881–897.

Yamauchi, K. (2001). Sequential learning and model selection with sleep. In L. Zhang, & F. Gu (Eds.), (*vol. 1*) (pp. 205–210). *ICONIP 2001 (Eighth international conference on neural information processing)*, Shanghai: Fudan University Press.

Yamauchi, K., Yamaguchi, N., & Ishii, N. (1999). Incremental learning methods with retrieving interfered patterns. *IEEE Transactions on Neural Networks*, *10*(6), 1351–1365.

Yanagawa, K., Fukushima, K., & Yoshida, T. (2002). Additional learnable neocognitron (in Japanese). Technical report of IEICE, No. NC2001-176.

Yingwei, L., Sundararajan, N., & Saratchandran, P. (1997). A sequential learning scheme for function approximation using minimal radial basis function neural networks. *Neural Computation*, *9*, 461–478.