Neocognitron Trained by Winner-Kill-Loser with Triple Threshold

Kunihiko Fukushima[1,2], Isao Hayashi[2], and Jasmin Léveillé[3]

[1] Fuzzy Logic Systems Institute, Iizuka, Fukuoka 820–0067, Japan
fukushima@m.ieice.org, http://www4.ocn.ne.jp/~fuku_k/index-e.html
[2] Faculty of Informatics, Kansai University, Takatsuki, Osaka 569–1095, Japan
[3] Department of Cognitive and Neural Systems and Center of Excellence for Learning in Education, Science, and Technology, Boston University, Boston, MA 02215, USA

Abstract. The *neocognitron* is a hierarchical, multi-layered neural network capable of robust visual pattern recognition. The neocognitron acquires the ability to recognize visual patterns through learning. The winner-kill-loser is a recently introduced competitive learning rule that has been shown to improve the neocognitron's performance in character recognition. This paper proposes an improved winner-kill-loser rule, in which we use a triple threshold, instead of the dual threshold used as part of the conventional winner-kill-loser. It is shown theoretically, and also by computer simulation, that the use of a triple threshold makes the learning process more stable. In particular, a high recognition rate can be obtained with a smaller network.

1 Introduction

The *neocognitron*, originally proposed by Fukushima [1], is a hierarchical, multi-layered neural network capable of robust visual pattern recognition. The neocognitron acquires the ability to recognize patterns through learning.

During learning, input connections to feature-extracting cells are modified upon presentation of training patterns. Several methods for training the neocognitron have been proposed to date. One of them, the *winner-kill-loser* learning rule, is known to be very powerful at training intermediate stages of the hierarchical network [2].

This paper proposes an improved learning rule, in which we use a triple threshold, instead of the dual threshold associated with the original winner-kill-loser. We show theoretically, and also by computer simulation, that the use of a triple threshold makes the learning process more stable.

2 Outline of the network

The neocognitron consists of layers of S-cells, which resemble simple cells in the 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.

The neocognitron discussed in this paper consists of four stages of S- and C-cell layers: $U_0 \rightarrow U_{S1} \rightarrow U_{C1} \rightarrow U_{S2} \rightarrow U_{C2} \rightarrow U_{S3} \rightarrow U_{C3} \rightarrow U_{S4} \rightarrow U_{C4}$. Here we use notation like U_{Sl}, for example, to indicate the layer of S-cells of the lth stage.

Each layer of the network is divided into a number of sub-layers, called *cell-planes*, depending on the feature to which cells respond preferentially. Incidentally, a cell-plane is a group of cells that are arranged retinotopically and share the same set of input connections [1]. As a result, all cells in a cell-plane have identical receptive fields but at different locations.

Stimulus patterns are presented to the input layer, U_0. The output of U_0 is then sent directly to U_{S1}. An S-cell in this layer resembles a simple cell in the primary visual cortex, and responds selectively to an edge at a particular orientation. To be more specific, U_{S1} has $K_{S1} = 16$ cell-planes, whose preferred orientations are chosen at an interval of 22.5°. As a result, contours in the input image are decomposed into edges for every orientation in U_{S1}. Unlike the S-cells in subsequent layers, S-cells in U_{S1} are made of analog threshold elements. Mathematically, an S-cell in U_{S1} extracts an oriented edge directly from U_0 using a linear filter followed by a half-wave rectification. The shape of the linear filter is encoded in the input connections to an S-cell and is implemented as a directional derivative of two-dimensional Gaussian. The neocognitron used here thus differs from previous versions [3] in which the S-cells were the same across all layers and where an additional contrast-extracting layer, U_G, was present between U_0 and U_{S1}.

S-cells at later stages (U_{S2} to U_{S4}) are each accompanied by an inhibitory V-cell. The V-cell, whose output is proportional to the root-mean-square of its input signals, inhibits the S-cell. In the conventional neocognitron, the V-cell performed a divisive normalization operation. In the current model, however, the V-cell inhibits the S-cell in a subtractive manner, which has been shown to increase robustness to background noise [4].

At each stage of the hierarchical network, the output of layer U_{Sl} is fed into layer U_{Cl}. C-cells have fixed input connections. By averaging their input signals, C-cells exhibit some level of translation invariance. As a result of averaging across position, C-cells encode a blurred version of their input. The blurring operation is essential for endowing the neocognitron with an ability to recognize patterns robustly, with little effect from deformation, change in size, or shift in the position of input patterns. Unlike previous versions of the neocognitron that used the arithmetic mean, in the current model averaging is done through root-mean-square [4].

As in previous versions of the neocognitron, excitatory connections to the C-cells in U_{C1} and U_{C2} are surrounded by inhibitory connections, yielding concentric on-center off-surround connections.

The strength of input connections to S-cells are modified through learning. After learning, S-cells become feature-extracting cells. S-cells at higher stages of the hierarchy extract more global features than S-cells at lower stages.

Learning is performed layerwise, from lower layers to higher layers, such that the training of a given stage can start only after the training of the preceding

stage is complete. In order to train S-cells in U_{Sl}, the responses of C-cells in the preceding layer U_{Cl-1} are used as a training stimulus. All S-cell layers, except for U_{S1}, are trained with the same training set.

We use the *winner-kill-loser* rule, a variant of competitive learning, to train intermediate layers U_{S2} and U_{S3} [2]. We propose to use a triple threshold to govern learning, instead of the dual threshold that was used in the original winner-kill-loser rule. This is the main topic of this paper and is discussed below in more detail.

As mentioned above, each layer of the neocognitron is divided into cell-planes. All cells in a cell-plane share the same set of input connections. This condition of shared connections has to be kept even during the learning phase, when input connections to S-cells are renewed. When a winner is chosen in a given cell-plane, its input connections are renewed based on the responses of the C-cells presynaptic to it. Since all cells in the cell-plane share the same set of connections, all other cells in the cell-plane come to have the same connections as the winner. The winner thus works like a seed in crystal growth. Hence we call it a *seed-cell*.

S-cells at the highest stage (U_{S4}) are trained by supervised competitive learning using labeled training data [3]. As the network learns varieties of deformed training patterns, more than one cell-plane per class is usually generated in U_{S4}.

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. However, if the winner has a wrong label (or if all S-cells are silent), a new cell-plane is generated. The new cell-plane hence learns the current training pattern simply by being assigned its corresponding label. Each cell-plane of U_{S4} thus has a label indicating one of the 10 digits.

During the recognition phase, the label of the maximally activated S-cell in U_{S4} determines the final result of recognition. The C-cells at the highest stage yield the inferred label of the input stimulus.

3 Competitive Learning with Winner-Kill-Loser

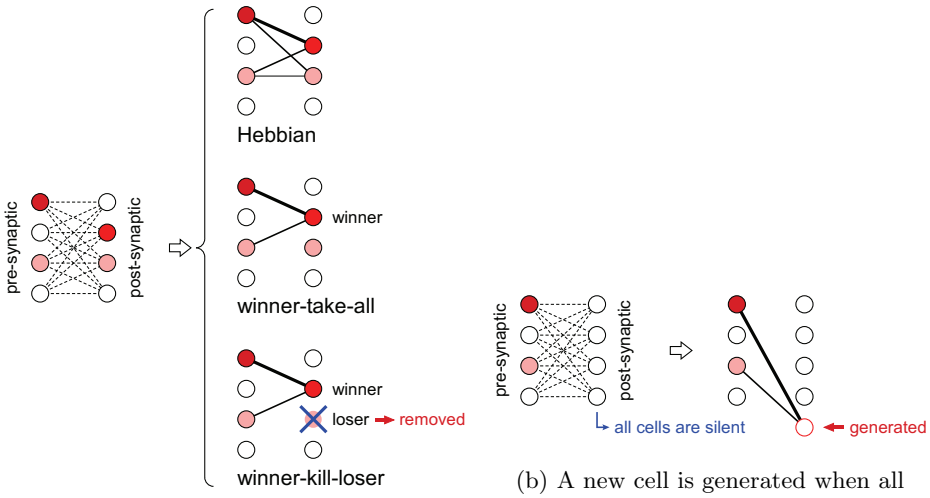
3.1 Winner-Kill-Loser with Dual Threshold

In order to train S-cells in layers U_{S2} and U_{S3}, we use competitive learning with winner-kill-loser [2]. We first explain the original winner-kill-loser rule, which uses a dual threshold for S-cells.

Fig. 1 illustrates the learning process with the original winner-kill-loser rule [2], and compares it with other learning rules.

The Hebbian rule, shown at the top of Fig. 1(a), is one of the most commonly used learning rules. During the learning phase, each synaptic connection is strengthened by an amount proportional to the product of the responses of the pre- and post-synaptic cells.

In the winner-take-all rule, shown in the middle of Fig. 1(a), post-synaptic cells compete with each other, and the cell from which the largest response is



(a) Several rules of learning.

(b) A new cell is generated when all postsynaptic cells are silent.

Fig. 1. Winner-kill-loser rule with dual threshold, in comparison with other learning rules. In this figure, the response of each cell is represented by the saturation of the color.

elicited becomes the winner. Only the winner can have its input connections renewed. The magnitude of the weight change is proportional to the response of the pre-synaptic cell. Incidentally, most of the conventional neocognitrons [1, 3] use this learning rule.

The winner-kill-loser rule, shown at the bottom of Fig. 1(a), resembles the winner-take-all rule in the sense that only the winner learns the training stimulus. In the winner-kill-loser rule, however, not only does the winner learn the training stimulus, but also the losers are simultaneously removed from the network. Losers are defined as cells whose responses to the training stimulus are smaller than that of the winner, but whose activations are nevertheless greater than zero.

If a training stimulus elicits non-zero responses from two or more S-cells, it means that the preferred features of these cells resemble each other, and that they work redundantly in the network. To reduce this redundancy, only the winner has its input connections renewed to fit more to the training vector, while the other active cells, namely the losers, are removed from the network.

Since silent S-cells (namely, the S-cells whose responses to the training stimulus are zero) do not join the competition, they are not removed. These cells are expected to work toward extracting other features.

As depicted in Fig. 1(b), a new S-cell is generated if all cells are silent for a given training stimulus. The initial value of the input connections of the newly generated S-cell is proportional to the response of the pre-synaptic cells. Incidentally, the generation of new S-cells was also a feature of the winner-take-all rule implemented in previous versions of the neocognitron.

In the learning phase, a number of training stimuli are presented sequentially to the network. During this process, generation of new cells and removal of redundant cells occurs repeatedly in the network. In particular, new cells are generated to cover areas of the multi-dimensional feature space that were not previously covered by existing cells. In the areas where similar cells exist in duplicate, redundant cells are removed. By repeating this process for a long enough time, the preferred features (reference vectors) of S-cells gradually become distributed uniformly over the multi-dimensional feature space.

When applying this learning rule to the neocognitron, a slight modification is required due to the fact that each layer of the network consists of a number of cell-planes such that all cells in a given cell-plane must share the same set of input connections both during learning and recognition.

At first, the S-cell whose response is the largest in the layer is chosen as a seed-cell. The seed-cell has its input connections renewed depending on the training vector presented to it. Once the connections to the seed-cell are renewed, all cells in the cell-plane from which the seed-cell is chosen come to have the same set of input connections as the seed-cell because of the shared connections. All non-silent cells at the same spatial location as the seed cell are determined losers, and the cell-planes to which they belong are removed from the layer.

In the original winner-kill-loser rule, a dual threshold is used to guide learning and recognition in S-cells [2]. Namely, S-cells have a higher threshold during learning (θ^{L}) than during recognition (θ^{R}). During learning, S-cells join the competition only when their responses to a training vector are not zero under the high threshold θ^{L}. This means that, even though an S-cell would yield a non-zero response under the low recognition threshold θ^{R}, that S-cell does not join the competition provided it is silent under the high learning threshold θ^{L}.

It has been demonstrated by computer simulation that the use of the winner-kill-loser rule in the competitive learning largely increases the recognition rate for smaller network sizes [2]. However, there still remain some problems in the winner-kill-loser with dual threshold.

Problems with the Dual Threshold Formulation As mentioned above, the original winner-kill-loser makes use of a dual threshold: a high threshold θ^{L} for learning and a low threshold θ^{R} for recognition. That is, in the learning phase, only one threshold, θ^{L}, is used.

We now use vector notation. Let $\boldsymbol{x}$ be the input signal from a set of presynaptic C-cells to an S-cell. We use vector $\boldsymbol{X}$ to represent the strength of input connections of the S-cell. We can interpret $\boldsymbol{X}$ as the preferred (optimal) feature of the S-cell in a multi-dimensional feature space. We sometimes call $\boldsymbol{X}$ the *reference vector* of the S-cell.

When a training vector $\boldsymbol{x}$ is presented, an S-cell calculates the similarity s between $\boldsymbol{X}$ and $\boldsymbol{x}$ by

$$s = (\boldsymbol{X}, \boldsymbol{x}) \,/\, \{\|\boldsymbol{X}\| \cdot \|\boldsymbol{x}\|\} \ . \tag{1}$$

If similarity s is larger than θ^{L}, the S-cell with subtractive inhibition yields a non-zero response [4]

$$u = \|\boldsymbol{x}\| \cdot \frac{\varphi[s - \theta^{\mathrm{L}}]}{1 - \theta^{\mathrm{L}}} \ , \tag{2}$$

where $\varphi[\ ]$ is a function defined as $\varphi[x] = \max(x, 0)$. The area that satisfies $s > \theta^{\mathrm{L}}$ in the multi-dimensional feature space is called the *tolerance area* of the S-cell. This situation is illustrated in Fig. 2.

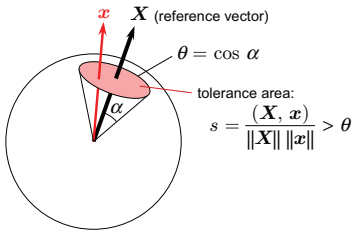


Fig. 2. Tolerance area of an S-cell in the multi-dimensional feature space.

If the response of the S-cell is the largest among all S-cells, the S-cell learns the training vector by adding $\boldsymbol{x}$ to $\boldsymbol{X}$. Other S-cells with non-zero response are determined losers and are removed from the network. If all S-cells are silent, a new cell is generated, and $\boldsymbol{x}$ becomes the reference vector of the generated S-cell.

It is expected that the learning process produces a situation where reference vectors of S-cells distribute uniformly in the multi-dimensional feature space as shown in Fig. 3(a).

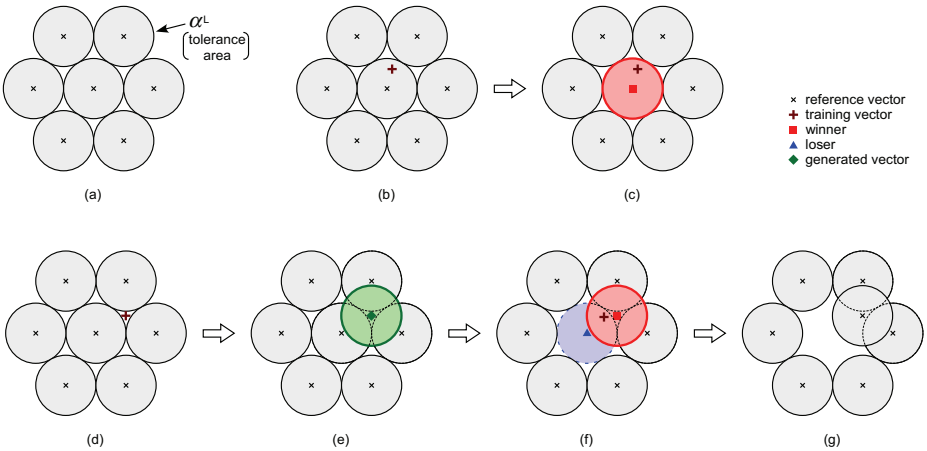


Fig. 3. Progress of learning by winner-kill-loser with a dual threshold. The dual threshold means that, in the learning phase, only one threshold, θ^{L}, is used.

Let us now observe how the distribution of reference vectors changes during learning. We assume that, at a certain moment in the learning, we happen to have a uniform distribution of reference vectors as shown in Fig. 3(a). If a training vector is presented at ✚ in (b) of the figure, the cell at ■ is the winner and learns the training vector as shown in (c). This is all right.

If a training vector is presented at ✚ as shown in (d), however, all cells are silent and a new cell, whose reference vector is at ◆, is generated as shown in (e). After that, if another training vector is presented at ✚ in (f), the cell at ■ becomes a winner and the cell at ▲ becomes a loser. Removal of the loser results in the situation depicted in (g). Thus, further training can actually destroy the desirable uniform distribution that was present in (a).

This means that removal and generation of cell-planes in the neocognitron do not stabilize during learning. Since the number of cell-planes continues to increase and decrease, the final number of cell-planes obtained after learning is determined largely by the duration of the learning phase. This in turn will strongly affect the network's recognition rate and scale.

3.2 Use of Triple Threshold for Winner-Kill-Loser

We now propose to add one more threshold during learning. In other words, we use thresholds θ^{W} and θ^{G}, instead of only one threshold θ^{L} (Fig. 4).

	dual threshold	triple threshold
learning phase	θ^{L}	θ^{W} for choosing a winner & losers
		θ^{G} for generating a new cell
recognition phase	θ^{R}	θ^{R}

Fig. 4. Comparison of dual and triple thresholds.

Threshold θ^{W} is used for determining the winner and losers. Non-zero responses are elicited from S-cells whose similarity s (namely, similarity between reference vector $\boldsymbol{X}$ of the cell and the training vector $\boldsymbol{x}$) is larger than θ^{W}. Among these S-cells, an S-cell that has the largest response becomes the winner and learns $\boldsymbol{X}$. Other non-silent S-cells are categorized as losers and are removed from the network.

The threshold θ^{G} works like a kind of subliminal threshold and controls the generation of a new S-cell (namely, the generation of a new reference vector in the vector space). If there exists at least one S-cell whose similarity s is larger than θ^{G}, no new S-cell is generated. In other words, not only an active S-cell that has $s > \theta^{\mathrm{W}}$, but also a silent S-cell whose similarity s is in the range $\theta^{\mathrm{W}} \geq s > \theta^{\mathrm{G}}$, can prevent the generation of a new S-cell. A new S-cell can be generated only when the similarity s of all S-cells are under the threshold θ^{G}. This situation is illustrated in Fig. 5.

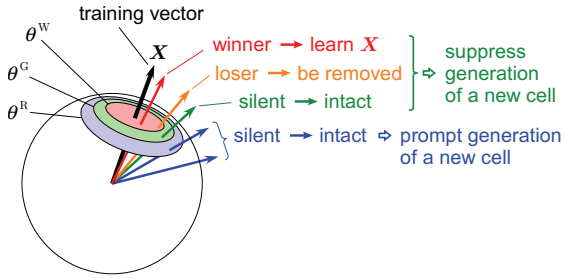


Fig. 5. Winner-kill-loser with triple threshold in the multi-dimensional vector space. Here we propose to use a subliminal threshold, θ^{G}, such that the presence of a cell whose similarity is greater than θ^{G} prevents the generation of a new cell.

Optimal Values of the Thresholds We now discuss how to choose threshold values θ^{W} and θ^{G}. To represent the radius of tolerance area, which is determined by threshold θ, we use angle α between two vectors in the multi-dimensional feature space, as shown in Fig. 2. Namely, $\theta^{W} = \cos\alpha^{W}$, and $\theta^{G} = \cos\alpha^{G}$. Although the feature space is actually of dimensionality greater than two, for simplicity we start our discussion assuming that it is a two-dimensional plane.

The goal of training is to make reference vectors distribute uniformly in the feature space. Once a desirable uniform distribution of reference vectors has emerged during learning, it should not be destroyed upon further training. To prevent reference vectors from becoming losers and being removed, the tolerance areas of radius α^{W} should not overlap (Fig. 6). In a feature space of dimensionality greater than one, however, it is inevitable that some vacant gaps are generated between non-overlapping disks of radius α^{W}. To prevent generation of a new reference vector, the feature space is covered by disks of radius α^{G} that can overlap with each other. The smallest α^{G} that can fill vacant gaps can be determined from α^{W}, as illustrated in the right of Fig. 6. Namely,

$$\alpha^{W} = \alpha^{G}\cos(\pi/6)\,, \qquad \text{or} \qquad \cos^{-1}\theta^{W} = \cos(\pi/6)\cos^{-1}\theta^{G}\,. \tag{3}$$

Simulations were conducted to assess the impact of the dual and triple thresholds in the neocognitron trained with the winner-kill-loser rule. The training set we use consists of 3000 handwritten digits (300 patterns for each digit) randomly sampled from the ETL1 database [5]. This training set is presented only once to train layers U_{S2} and U_{S3}. Fig. 7 shows how the number of cell-planes in layer U_{S3} changes during learning. Results for the triple and dual thresholds are depicted as a red solid line and a blue dotted line, respectively. It can be seen from the figure that the fluctuation of the number of cell-planes is much smaller with the triple threshold and that the learning can progress more stably. The final number of cell-planes that has been created after learning is usually smaller with the triple threshold than with the dual threshold.

Fig. 8 shows how the error rate of the neocognitron changes with the size of the training set. The test set consists of 5000 patterns (500 patterns for each

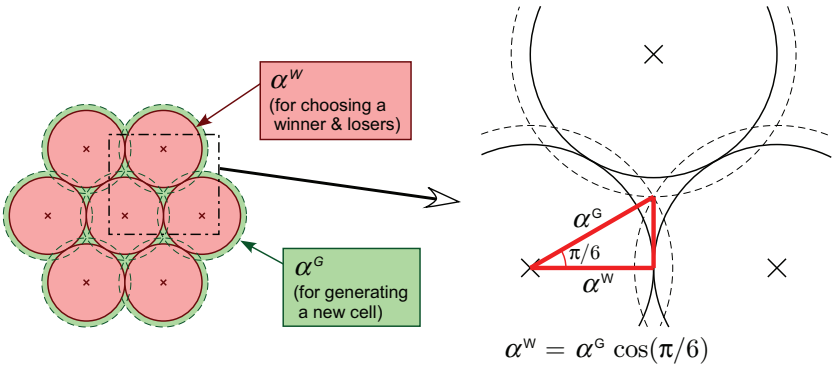


Fig. 6. Winner-Kill-Loser with triple threshold. Use of thresholds $\theta^{\mathrm{G}}(= \cos\alpha^{\mathrm{G}})$ and $\theta^{\mathrm{W}}(= \cos\alpha^{\mathrm{W}})$ for the learning.

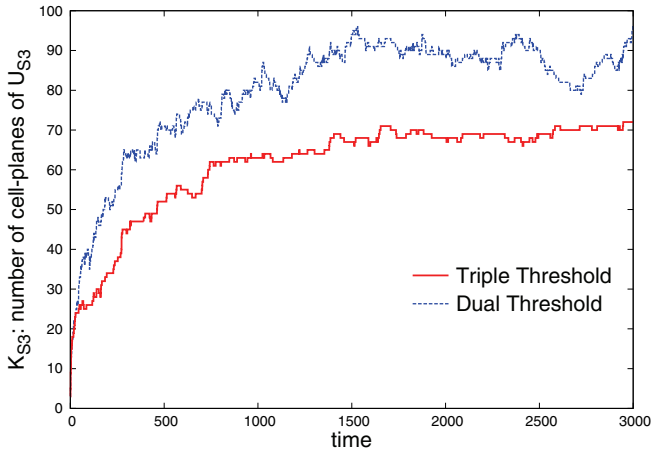


Fig. 7. Number of cell-planes (K_{S3}) during the learning.

digit). Experiments were repeated twice for each condition, using different learning and test sets randomly sampled from the ETL1 database [5], and the results were averaged across the two experiments. We can see that the recognition error itself does not differ much whether using the dual (blue dotted line) or triple threshold (red solid line). Although the recognition error of the neocognitron depends on the final number of cell-planes that have been created after learning, we usually have almost the same recognition rate with a smaller number of cell-planes when the network is trained with the triple threshold.

In the case of the result shown in Fig. 7, for example, when using the triple threshold, the final number of cell-planes in each layer was $(K_{\mathrm{S2}}, K_{\mathrm{S3}}, K_{\mathrm{S4}}) = (31, 72, 73)$, and the recognition error was 1.22%. On the other hand, under the dual threshold, we had $(K_{\mathrm{S2}}, K_{\mathrm{S3}}, K_{\mathrm{S4}}) = (30, 96, 82)$ and 1.26%, respectively.

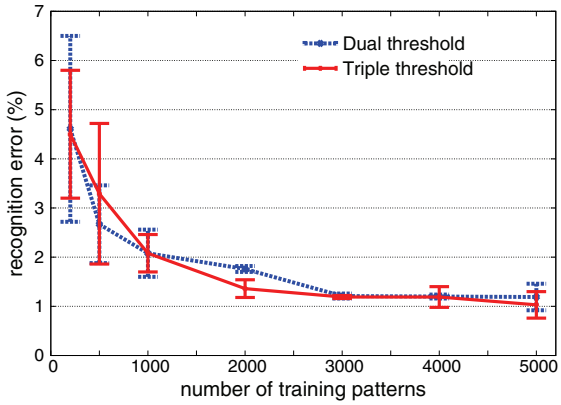


Fig. 8. Recognition error vs. number of training patterns.

It should be noted here that the computational cost for calculating the response of U_{Sl} is approximately proportional to $K_{Sl-1} \times K_{Sl}$.

4 Discussions

In this paper we introduce a new triple threshold to be used for competitive learning with the winner-kill-loser rule. We show by computer simulation that the use of the triple threshold makes the learning process more stable than when using the dual threshold. Although the triple threshold does not seem to improve recognition rate, it nevertheless significantly reduces network scale (with a smaller number of cell-planes in each layer). One of the greatest merits of the triple threshold formulation is the stability of learning, which in turn makes the neocognitron less sensitive to the duration of the learning phase.

References

1. Fukushima, K.: Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics, 36(4)*, 193–202 (1980).
2. Fukushima, K.: Neocognitron trained with winner-kill-loser rule. *Neural Networks, 23(7)*, 926–938 (2010).
3. Fukushima, K.: Neocognitron for handwritten digit recognition. *Neurocomputing, 51*, 161–180 (2003).
4. Fukushima, K.: Increasing robustness against background noise: visual pattern recognition by a neocognitron. *Neural Networks*, in print (2011). doi:10.1016/j.neunet.2011.03.017
5. ETL1 database: `http://www.is.aist.go.jp/etlcdb/#English`