# Associative Learning in Hierarchical Self-Organizing Learning Arrays

Janusz A. Starzyk, *Senior Member, IEEE*, Zhen Zhu, *Student Member, IEEE*, and Yue Li

***Abstract*—In this paper, we introduce feedback-based associative learning in self-organized learning arrays (SOLAR). SOLAR structures are hierarchically organized networks of sparsely connected neurons that define their own functions and select their interconnections locally. This paper provides a description of neuron self-organization and signal processing. Feedforward processing is used to make necessary correlations and learn the input patterns. Discovered associations between neuron inputs are used to generate feedback signals. These feedback signals, when propagated to the primary inputs, can establish the expected input values. This can be used for heteroassociative (HA) and autoassociative (AA) learning and pattern recognition. Example applications in HA learning are given.**

***Index Terms*—Associative learning, feedback structure, pattern recognition, self-organizing learning array (SOLAR).**

## I. Introduction

ASSOCIATIVE learning is one of the necessary elements of intelligence, thus it is desirable that an artificial system that mimics biological intelligence be able to perform both spatial and temporal associations. Associative networks were developed as a special class of artificial neural networks (ANNs) to handle associative learning and retrieval of information. There are two types of associative networks: heteroassociative (HA) and autoassociative (AA). HA networks are capable of making associations between two or more different types of input signals. For instance, a HA network may associate a verbal command with an image or text. AA networks learn associations between elements of the same input vector. Such a network can learn various patterns, and then recall the pattern based on a fractional part. For instance, it could recall an original pattern based on a corrupt or partially missing pattern. Examples of HA networks include the multilayer perceptron [1], the counterpropagation network [2], the bidirectional associative memory [3], and multiassociative spatiotemporal network [4], whereas the Hopfield network [5] and the Vogel associative memories [6], [7] are AA. Among these algorithms, the bidirectional associative memory has received more attention in the recent years on its noise performance [8], [9] and stability [10]. Its variation designed for HA learning [11], [12] can accomplish one-to-many association. In this paper, we present a model of the self-organizing learning array (SOLAR) that implements both HA and AA learning.

Spatiotemporal associations are particularly important in both biological and electromechanical systems. For example, a spatiotemporal association may trigger a reactive response in an animal or guide the robot to its target. Time delays have been used in Hopfield networks [5] to generate spatiotemporal sequences which are time-dependent sequences of spatial patterns. Storage and retrieval of spatiotemporal sequences have been studied in many papers (see, e.g., [13]). Buhmann [14] used stochastic noise to link spatial patterns in Hopfield networks and obtain spatiotemporal sequences, which use Hebbian learning rules [15] and, therefore, have relatively low memory capacity. Guyon [16] presented a spatiotemporal system with analytical expressions to represent stored sequences. Many other mechanisms for spatiotemporal sequence generation have been developed. For instance, Wang [17] proposed a network to generate complex spatiotemporal sequences based on symbolic inputs to represent associated patterns, whereas Wang [18] proposed multiassociative neural networks for learning and retrieving spatiotemporal patterns using the counterpropagation mechanism with delay lines.

While the proposed approaches achieved reasonable storage and retrieval of input sequences, they have some serious drawbacks if one wants to implement them in biologically plausible structures. The problem with counterpropagation networks or networks with Hebbian learning is that they require global wiring to all the pattern inputs from each competing neuron. Thus, the number of interconnection wires grows quadratically with the number of patterns stored, and such networks do not scale well. In this paper, we take a different approach to pattern storage and associations. A hierarchical, multilayer structure based on our self-organizing learning architecture [19] is used, and we will demonstrate that such a structure can make the necessary associations between patterns using sparsely connected neurons.

Prior study of SOLAR structures reported in [19] concentrated on demonstrating its pattern recognition and classification abilities. In this paper, a feedback mechanism with inhibitory connections and associative learning to SOLAR is introduced and its performance on selected machine learning problems is demonstrated. By comparing the activation levels of the input neurons triggered by the associative feedback activities, we can determine whether SOLAR made correct associations, and for example, correctly recognized an observed pattern.

Manuscript received May 10, 2005; revised December 30, 2005.

J. A. Starzyk and Z. Zhu are with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH 45701 USA (e-mail: starzyk@bobcat.ent.ohiou.edu).

Y. Li was with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH 45701 USA. She is now with O2Micro, Inc., Santa Clara, CA 95054 USA.

Color versions of Figs. 6–9 and 11–13 are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2006.883008

This paper has been organized into five sections. Section II discusses the structure of the proposed network, followed by a description of network behavior in Section III. Section IV presents testing results on several bench-mark machine learning problems. Section V presents conclusion with ideas for future work.

## II. SOLAR

### A. Properties of SOLAR

SOLAR is a regular, two-dimensional (2-D) or three-dimensional (3-D) array of identical processing cells connected to programmable routing channels, as introduced in [19]. Each cell in the array has the ability to self-organize by adapting its functionality and interconnection structure in response to information contained in its input signals.

Like ANNs, SOLAR is inspired by the structure of biological neural networks and shares their robust, distributed, and parallel signal processing, yet it differs from existing realizations of ANNs. It has a deep multilayer hierarchical structure, which helps to handle the complexity of target problems. It uses online learning with dynamically set neuron functions and dynamically learned sparse connections that are efficient in hardware realization. The structure is compatible with the self-organization of cortical neurons [20], where half of initially grown interconnections between neurons are pruned between early childhood and adolescence.

A SOLAR structure in many ways resembles the organization of cellular neural networks (CNNs). As in a CNN, the architecture is defined by an array of identical cells, which adapt their behavior to input data. The neurons are cellular automata, which can be programmed to perform different computational tasks based on data received from neighbors. Neurons can be either static or dynamic, depending on their implementation and types of signals processed. However, unlike a CNN, the connectivity structure is not fixed. The interconnection structure of a CNN is defined by templates, which limits its learning ability, whereas the interconnection structure of a SOLAR can be dynamically changed during the network's operation, and contributes to learning. Thus, a CNN can be considered a special case of the SOLAR structure.

An ANN forces specific output neurons to the trained values, minimizing the error between the desired and the actual responses. This is not a natural response mechanism that may be observed in the real brain. When the visual cortex recognizes a specific object, a single neuron or a group of neurons may fire somewhere in the middle of the brain structure, rather than at a specified output location. How does the brain manifest its recognition of the object? The process that follows this recognition is complex and the brain does not immediately notify an observer that recognition took place. The brain thinks about the consequences of this recognition and occasionally the result of such thinking manifests itself through action (activation of motor neurons). This is when we can observe that the desired information was recognized and acted upon.

In SOLAR, an object is also recognized internally and a neuron(s) responsible for recognition is not wired to an output to be observed. In fact, it is impossible to predict where the recognition will take place, so we must develop a scheme to verify the correctness of this recognition no matter which neuron is responsible for it. One way of checking correctness of pattern recognition is to use association. A simple pattern code is used and associated with each class of the target patterns (a class ID in classification applications.) This pattern code is presented to the SOLAR structure together with the target pattern to be memorized.

All that the neurons need to do is learn associations between these two patterns, representing the local features and the classification ID, respectively. Once the learning is completed, a network is capable of making the necessary associations, such that when presented with the target pattern only, it drives the associated input signals to the values that represent the class ID pattern. The network uses feedback signals to pass information backwards to the associated inputs. No backpropagation of error takes place, and only feedforward learning is used. Feedback lines activate input code neurons when activity on associated neurons is detected. By using this associative learning scheme, a network is able to discover the inner correlation between the inputs and recover absent or uncertain portions of these inputs. The network behaves in the same way when it is presented with only a portion of the target pattern and is used to recover the missing portion of the pattern. Similar to the recognition application, missing data can be found from feedback traced to the unknown inputs.

### B. Network Structure

According to its definition [19], a SOLAR network can be formed as a regular 2-D or 3-D lattice of identical neurons. In a 2-D structure, the number of rows defines the input span. For a recognition task, it is set equal to or greater than the dimensionality of the pattern inputs plus the associated class ID pattern inputs. The number of columns indicates the depth of the network (number of hierarchical levels), which is set according to the input span. In a 3-D structure, the input span is 2-D with the third dimension representing the hierarchical processing depth. The network is pseudorandomly constructed with an interconnection structure of small world networks [21]. A 3-D network is expected to handle image related tasks better. It is believed that biological neurons tend to have mostly local connections [22], and therefore, the neurons in this associative learning network also have a larger chance to be connected to neurons within a short Euclidean distance. Both the regular array structure and sparse local wiring are favored by hardware implementation, both in field-programmable gate arrays (FPGAs) and analog very large scale integration (VLSI) implementation platforms. This structure fits silicon chip interconnection strategy better and the sparse wiring saves design area when compared to the intensive interconnections required by ANNs.

In general, SOLAR neurons can be connected to many other neurons and their connectivity can be dynamically learned [23]. Since the major concentration of this work is on associative learning, we have assumed a fixed interconnection structure for simplicity of discussion. Furthermore, in the current implementation, each neuron has only two inputs $I_1$ and $I_2$, received from

the outputs of other neurons or from the sensory inputs, and a single output $O$. This makes it more convenient to design the neural transfer functions and the associative learning scheme. Since binary trees of such neurons can implement networks capable of arbitrarily complex associations, two-input neurons do not limit complexity of the network operation. Network operations have been designed based upon such an interconnection structure, which is not a restriction to our model.

## III. Network Operations

### A. Signal Definition and Data Preparation

In the associative learning model implemented in this paper, outside input is presented to the network in a binary form. Details of signal digitization will be discussed in Section IV. The internal signals of the network have semilogic values ranging from 0 to 1, where 0 and 1 correspond to logic false or true, respectively. A signal is determinate low or determinate high if its value is 0 or 1. It is low if it is below 0.5 but above 0, and high if it is above 0.5 and below 1. The signal strength is measured as the distance between the signal level and 0.5. The "inactive" input of unknown logic is 0.5. These semilogic signals can also be understood as the likelihood of being determinate logic values. These signals will be further explained and applied in the feedback calculation in Section III-D.

### B. Associative Learning

The task of a neuron during training is to discover the potential relationship between its two inputs and to remember it. If a neuron is able to observe any correlations in the statistics of the input connections, it will function as an associative neuron. Otherwise, it will be a transmitting neuron.

Since the signals have been defined in a semilogic format, the probabilities of $I_1$ and $I_2$ being "low" or "high" and their joint probabilities can be recorded using counters located in each neuron. The conditional probabilities $P(I_2|I_1)$ and $P(I_1|I_2)$ can then be computed, for example, $P(I_2 \text{ is low}|I_1 \text{ is low}) = P(I_2 \text{ is low and } I_1 \text{ is low})/P(I_1 \text{ is low})$. Similarly, $P(I_2 \text{ is high}|I_1 \text{ is low})$, $P(I_2 \text{ is low}|I_1 \text{ is high})$, and $P(I_2 \text{ is high}|I_1 \text{ is high})$ can be easily obtained.

A simplified confidence interval measure is used for each of the probabilities: $\text{CI} = 2(1 - P(I_2|I_1))/\sqrt{N}$ where $N$ stands for the number of training inputs encountered. A confidence interval helps to ensure that the estimated probabilities stay above a specified threshold. The value of $P(I_2|I_1) - \text{CI}$ is then compared against a threshold $\tau$, where $\tau = 0.9$ was used in this paper. With $P(I_2 \text{ is low}|I_1 \text{ is low}) - \text{CI} > \tau$, we say that $I_2$ can be **implied** from $I_1$. Likewise, $P(I_1|I_2)$ decides whether $I_1$ can be **implied** from $I_2$.

*Definition:* Neuron inputs $I_1$ and $I_2$ are **associated** if and only if $I_2$ can be implied from $I_1$ and $I_1$ can be simultaneously implied from $I_2$. Such a neuron is then an **associative neuron**.

Fig. 1 shows an example of the $I_1$ and $I_2$ inputs an associative neuron has received in training. Clearly, $I_1$ and $I_2$ are most likely to be simultaneously low or high although there is some noise. This can be verified using $P(I_2|I_1)$ and $P(I_1|I_2)$.

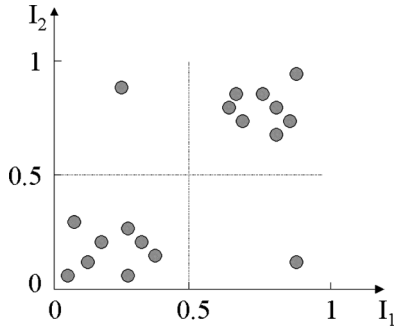



Fig. 1. Input distribution in an associative neuron.

### C. Feedforward Scheme

Based on the data distribution, the neuron needs to select a proper transfer function $f$ from a predefined set $F$ that can best describe the relationship between $I_1$ and $I_2$. The neuron's output can then be generated using $f$. For the neuron shown in Fig. 1, $I_1$ and $I_2$ usually become simultaneously low or high, so $f$ can be defined in the following way:

$$f(I_1, I_2) = \begin{cases} 0, & \text{if } I_1 \text{ is low} \\ 1, & \text{if } I_1 \text{ is high.} \end{cases} \tag{1}$$

The neuron output is set "inactive" or 0.5, whenever either one of the inputs is 0.5

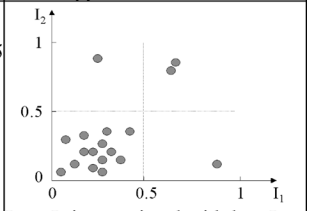
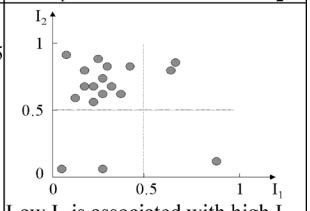
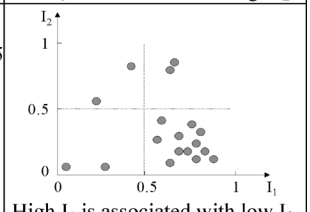
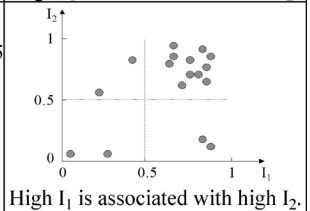
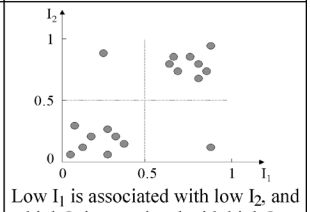
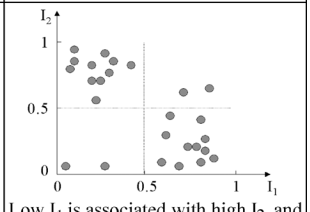
$$O = \begin{cases} 0.5, & \text{if } I_1 = 0.5 \text{ or } I_2 = 0.5 \\ f(I_1, I_2), & \text{otherwise.} \end{cases} \tag{2}$$

$F$ is composed of six functions that are designed to include all the logic relationships between $I_1$ and $I_2$ in an associative neuron. Using the semilogic signals with continuous values, the relationships can be observed from the training data distribution in the $I_1/I_2$ space. In an associative neuron, the majority of the training data is distributed in one of the $I_1/I_2$ quadrants or two diagonal quadrants. If a quadrant (e.g., $I_1 < 0.5$ and $I_2 < 0.5$) holds most of the data points, it is called a **dominant quadrant**. The relationship between $I_1$ and $I_2$ in Table I is described through each function's **applicable conditions**. For example, "low $I_1$ is associated with low $I_2$," is the applicable condition for $f_1$. Functions $f_2$ through $f_4$ all have their applicable conditions described in a similar way.

The outputs of $f_1$ to $f_4$ are always 1 for any data point located in the dominant quadrant, and 0 for all the others. When data points are mostly distributed in two diagonal quadrants, $I_1$ and $I_2$ can either be synchronously low or high, as shown in Fig. 1 (function $f_5$), or complement each other, as in $f_6$. To accommodate noise, the outputs of $f_5$ and $f_6$ are defined arbitrarily based only on $I_1$. All the functions are listed in Table I together with their applicable conditions. Input distribution layout figures similar to Fig. 1 are used to illustrate each of the conditions. These functions benefit the feedback scheme, which results in identification or recovery of associated inputs.

If the training data that the neuron receives shows any distribution other than what is included in Table I, association between $I_1$ and $I_2$ cannot be established and the neuron simply chooses the input with higher entropy (dominant input) and transmits this input value to its output. Such a neuron is called

TABLE I
FUNCTIONS AND THEIR APPLICABLE CONDITIONS

| Function | Applicable conditions |
|---|---|
| $f_1(I_1,I_2)=\begin{cases}1, \text{if } I_1<0.5 \,\&\, I_2<0.5\\0, \text{ else}\end{cases}$ |  Low $I_1$ is associated with low $I_2$. |
| $f_2(I_1,I_2)=\begin{cases}1, \text{if } I_1<0.5 \,\&\, I_2>0.5\\0, \text{ else}\end{cases}$ |  Low $I_1$ is associated with high $I_2$. |
| $f_3(I_1,I_2)=\begin{cases}1, \text{if } I_1>0.5 \,\&\, I_2<0.5\\0, \text{ else}\end{cases}$ |  High $I_1$ is associated with low $I_2$. |
| $f_4(I_1,I_2)=\begin{cases}1, \text{if } I_1>0.5 \,\&\, I_2>0.5\\0, \text{ else}\end{cases}$ |  High $I_1$ is associated with high $I_2$. |
| $f_5(I_1,I_2)=\begin{cases}0, \text{if } I_1<0.5\\1, \text{if } I_1>0.5\end{cases}$ |  Low $I_1$ is associated with low $I_2$, and high $I_1$ is associated with high $I_2$. |
| $f_6(I_1,I_2)=\begin{cases}0, \text{if } I_1<0.5\\1, \text{if } I_1>0.5\end{cases}$ |  Low $I_1$ is associated with high $I_2$, and high $I_1$ is associated with low $I_2$. |

a **transmitting neuron**. An input $I_1$ of a transmitting neuron is called the **dominant input** if

$$\left|P(I_1 \text{ is low})-P(I_1 \text{ is high})\right|<\left|P(I_2 \text{ is low})-P(I_2 \text{ is high})\right|. \tag{3}$$

In a transmitting neuron, the dominant input is directly linked to the output, while the other input is ignored. The neuron acts as a dominant input transmitter.

### D. Feedback Scheme

During testing, missing parts of the data need to be recovered from existing data through association. For example, in a pattern recognition problem, the associated class ID pattern inputs are unknown and, therefore, are set to 0.5. The neurons that are physically connected to these inputs use feedback from the associative neurons to define these unknown input values. This, in turn, can be used either to classify the input pattern or to recover the uncertain part of the input.

The feedback scheme is an important element of the associative learning in a SOLAR network. A neuron's output is linked to the inputs of multiple neurons, as can be seen in Fig. 2. A neuron generates its output $O$ from its feedforward operation (Table I), and also receives different feedback from the connected neurons.

The following describes how the feedback is generated and transmitted through the network. A signal's strength in this network is measured relative to 0.5. When the generated output $O$ is weaker than the strongest feedback output $O_f$ (called a **dominant feedback**) it may trigger the feedback to the neuron's inputs. The input/output relationship in the feedback scheme for each neuron can be described by one of the four following scenarios. In the following discussion, we assume that a neuron received a feedback signal from one or more of its output neurons. To illustrate how this is done, consider the following example. The general rules are as follows.

*Example 1:* Assume that the original inputs of neuron $\mathbf{N_3}$ are 0.5 and 1, respectively. Then, according to (2), the generated output is $O=0.5$, independent of the type of function used. The strongest output feedback that $\mathbf{N_3}$ receives is $O_f=O_{f2}=0.9$. It triggers the feedback scheme inside the neuron. As a result of training, neuron $\mathbf{N_3}$ uses function $f_1$. In this case, $O_f>0.5$ indicates that the input to this neuron should most likely be in the quadrant $I_1<0.5$ and $I_2<0.5$. The feedback input can thus be calculated as $I_{1f}=1-O_f=0.1$ and $I_{2f}=1-O_f=0.1$. Details of the feedback evaluation for different transfer functions are discussed in Section III-E. $I_{1f}$ is stronger than $I_1=0.5$ and $I_{2f}$ is weaker than $I_2=1$. Therefore, $I_1$ is set to $I_{1f}$ and passed back onto $\mathbf{N_1}$, while $I_{2f}$ has no effect.

Notice that, in general, the two inputs of a neuron can be changed and they may generate a different output for the connecting neurons. However, this updated output value should not be allowed to propagate forward to higher hierarchical layers and, thus iterative processes can be avoided. In particular, outputs $O_{f1}$ and $O_{f3}$ of $\mathbf{N_3}$ shown in Fig. 2 cannot be updated with $O_{f2}$, since this may cause instability and start incorrect firing through the entire network. Therefore, the dominant feedback signal $O_{f2}$ plays an additional role similar to that of an inhibitory synapse in biological neural networks. Notice that weak and yet active signals, which have values other than 0, 1 and 0.5, can be generated only in the output feedback. The inhibitory connections prevent such weak signals from becoming inputs to other neurons.

1) *Transmitting neurons:* A transmitting neuron simply passes $O_f$ back to its dominant input. When the feedback $I_{1f}$ is stronger than the dominant input $I_1$ (e.g., neuron $\mathbf{N_1}$ in Fig. 2) $I_1$ will be overwritten by $I_{1f}$. Note that $I_1=0.5$ may come from unknown sensory inputs.

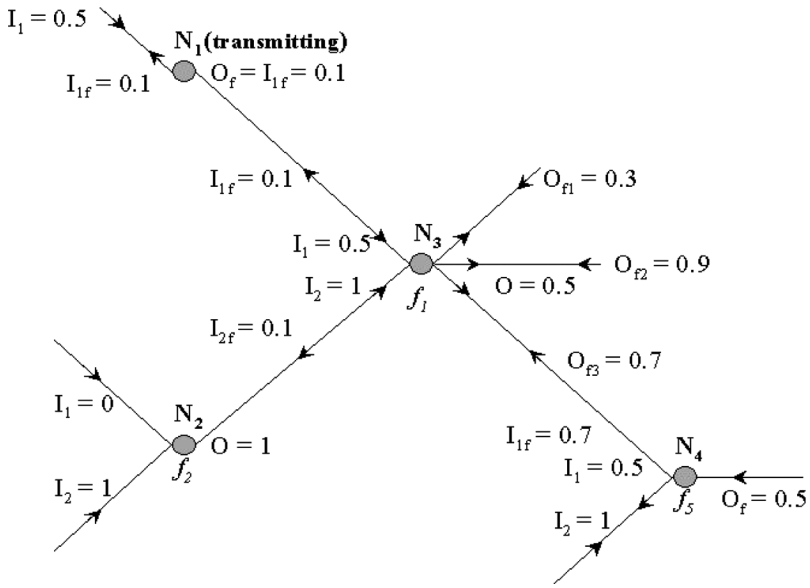



Fig. 2. Neuron feedback scheme.

2) *Associative neurons with strong inputs:* If $I_1$ or $I_2$ of an associative neuron (e.g., neuron $\mathbf{N_2}$ in Fig. 2) are both at full strength, (either 0 or 1) the generated output $O$ will be at full strength. The output $O$ of $\mathbf{N_2}$ can be computed using transfer function $f_2$ defined in Table I. Feedback has no effect and this neuron only passes information forward. These neurons are very likely connected to known data inputs.
3) *Associative neurons with active feedback and inactive input(s):* For an associative neuron that does not have determinate signals on both inputs (e.g., neuron $\mathbf{N_3}$ in Fig. 2) $O_f$ creates feedback $I_{1f}$ and $I_{2f}$ through the function $f$. If the feedback signals are stronger than the original inputs $I_1$ and $I_2$, these inputs will be overwritten. Consequently, overwritten inputs become feedback signals to the output of neurons $\mathbf{N_1}$ and $\mathbf{N_2}$, to which $\mathbf{N_3}$ inputs are linked. This scenario was illustrated in Example 1.
4) *Associative neurons with inactive feedback and one inactive input:* Some associative neurons located deeply in the network may not receive active feedback (e.g., $\mathbf{N_4}$ shown in Fig. 2). If one of their input connections is inactive, they will pass the information from the known input to the unknown input through association. The inactive input connection will be overwritten based on the other input and the neuron function $f$. Since $I_2$ is 0.5 for $\mathbf{N_4}$, the feedback to $I_2$ is determined from the known input $I_1$ and the transfer function $f_5$. Neurons of this type are called **actively associating** and are the backbone of the associative processing in SOLAR.

TABLE II
FEEDBACK CALCULATION

| $f$ | $I_{1f}$ | $I_{2f}$ |
|---|---|---|
| $f_1$ | $I_{1f} = \begin{cases} 1-O_f, & \text{if } O_f > 0.5 \\ A(I_2), & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} 1-O_f, & \text{if } O_f > 0.5 \\ A(I_1), & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |
| $f_2$ | $I_{1f} = \begin{cases} 1-O_f, & \text{if } O_f > 0.5 \\ A(1-I_2), & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} O_f, & \text{if } O_f > 0.5 \\ A(1-I_1), & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |
| $f_3$ | $I_{1f} = \begin{cases} O_f, & \text{if } O_f > 0.5 \\ A(1-I_2), & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} 1-O_f, & \text{if } O_f > 0.5 \\ A(1-I_1), & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |
| $f_4$ | $I_{1f} = \begin{cases} O_f, & \text{if } O_f > 0.5 \\ A(I_2), & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} O_f, & \text{if } O_f > 0.5 \\ A(I_1), & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |
| $f_5$ | $I_{1f} = \begin{cases} w_5, & \text{if } O_f > 0.5 \\ 1-w_5, & \text{if } O_f < 0.5 \\ I_2, & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} w_5, & \text{if } O_f > 0.5 \\ 1-w_5, & \text{if } O_f < 0.5 \\ I_1, & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |
| $f_6$ | $I_{1f} = \begin{cases} w_6, & \text{if } O_f > 0.5 \\ 1-w_6, & \text{if } O_f < 0.5 \\ 1-I_2, & \text{if } I_1 = 0.5 \\ I_1, & \text{otherwise} \end{cases}$ | $I_{2f} = \begin{cases} 1-w_6, & \text{if } O_f > 0.5 \\ w_6, & \text{if } O_f < 0.5 \\ 1-I_1, & \text{if } I_2 = 0.5 \\ I_2, & \text{otherwise} \end{cases}$ |

### E. Feedback Evaluation

For neurons that fit scenarios 3) and 4), feedback is defined for each function as illustrated in Table II. For functions $f_1$ through $f_4$ with $O_f > 0.5$, both input feedback signals can be directly determined. For $O_f < 0.5$, the stronger of $I_1$ or $I_2$ will be used to determine the feedback of the other. Neurons that fit scenario 4) always have $O_f = 0.5$ and have to determine feedback to the weak input from the strong one. Detailed feedback calculations are listed in Table II and are based on the function type and the signal strength.

In Table II, function $A(x)$ determines the signal strength of $x$ when a neuron is using $f_1$ through $f_4$ with $O_f <= 0.5$. $A(x)$ can be defined for $f_1$ as follows:

$$A(I_2) = \begin{cases} A'(w_1), & \text{if } O_f < 0.5, I_2 > 0.5 \\ 0.5, & \text{if } O_f < 0.5, I_2 = 0.5 \\ 1 - A'(I_2), & \text{if } O_f < 0.5, I_2 < 0.5 \\ w_1, & \text{if } O_f = 0.5, I_2 > 0.5 \\ 0.5, & \text{if } O_f = 0.5, I_2 = 0.5 \\ I_2, & \text{if } O_f = 0.5, I_2 < 0.5 \end{cases} \quad (4)$$

where $w_1$ is the likelihood of an input being high under the feedback condition

$$w_1 = \frac{P(I_1 \text{ is high and } I_2 \text{ is high})}{P(I_1 \text{ is high and } I_2 \text{ is high}) + P(I_1 \text{ is low and } I_2 \text{ is high})} \quad (5)$$

Function $A'(x)$ defined as $A'(x) = 0.5 - 2 \cdot |(O_f - 0.5) \cdot (0.5 - x)|$ attenuates signal $x$ using the strength of $O_f$ measured as $2 \cdot |O_f - 0.5|$. With $O_f < 0.5$, a neuron using $f_1$ through $f_4$ knows that the data point is not in the dominant quadrant. Based on this fact, $I_{1f}$ can be estimated from $I_2$ and the likelihood value $w_1$, combined with $A'(x)$.

1) For the first condition of (4), $w_1$ gives the likelihood that $I_{1f}$ is greater than 0.5. If $w_1$ is greater than 0.5, $I_{1f}$ is most likely high; otherwise, it is most likely a low signal.
2) For the second condition of (4), $I_{1f} = 0.5$ if $I_2 = 0.5$.
3) For the third condition, when $I_2$ is less than 0.5 and the data point is not in the dominant quadrant of $f_1$ where both inputs are low, the only possibility is that $I_{1f}$ is high. Its strength is then determined based on $I_2$, but is attenuated according to $O_f$.
4) For the fourth condition, $I_{1f}$ is estimated from $w_1$ in a manner similar to that of the first condition. Notice that with $O_f = 0.5$, the neuron is actively associating and $w_1$ needs no attenuation.
5) For the fifth condition, $I_{1f} = 0.5$ if $I_2 = 0.5$.
6) For the sixth condition, the neuron determines $I_{1f}$ from $I_2$ and since $I_2$ is low, $I_{1f}$ is also low.

For function $f_1$, $A(I_1)$ can be defined symmetrically to $A(I_2)$. Things are slightly different for $f_2$ through $f_4$, as the dominant quadrant is different for each of the functions. All the versions of $A(I_1)$ and $A(I_2)$ can be defined following the principle described previously (details not given because of space constraints). Functions $f_5$ and $f_6$ are defined based on two diagonal quadrants, as can be seen from Table I. Using the definitions of $f_5$ and $f_6$, their feedback signals $I_{1f}$ and $I_{2f}$ can be directly determined from $O_f$. However, the signal strength of $I_{1f}$ and $I_{2f}$ needs to be computed from the likelihood measurements $w_5$ and $w_6$ defined in (6) and (7) for $f_5$ and $f_6$, respectively

$$w_5 = P(I_1 = \text{high}, I_2 = \text{high}) + P(I_1 = \text{low}, I_2 = \text{low}) \quad (6)$$

$$w_6 = P(I_1 = \text{low}, I_2 = \text{high}) + P(I_1 = \text{high}, I_2 = \text{low}). \quad (7)$$

The values $w_5$ and $w_6$ are the probabilities that a data point can be located within the expected quadrants. They estimate the likelihood that each feedback can be correctly generated from a neuron's associative learning experience. In a manner similar to the use of $w_1$, likelihood measures such as $w_5$ and $w_6$ are used to evaluate the signal strength of $I_{1f}$ and $I_{2f}$ for $f_5$ and $f_6$.

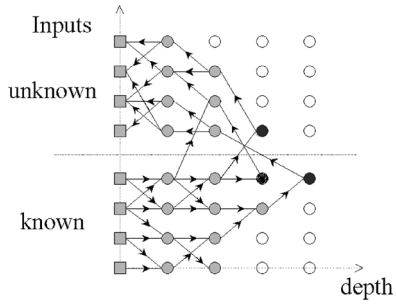



Fig. 3. Network structure.

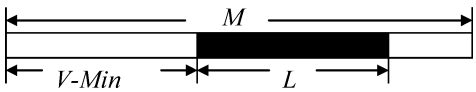



Fig. 4. $M$ input bits sliding bar code.

With $I_{1f}$ and $I_{2f}$ found for each neuron, the working mechanism of the whole associative learning network can be illustrated using the diagram in Fig. 3.

The gray rectangles in Fig. 3 stand for the sensory inputs including known and unknown input values. The white circles are the neurons that do not participate in the input signal feedforward or feedback processing (with unknown inputs and outputs) and their connections are not displayed in this figure for simplicity. The black circles are actively associating neurons. The gray circles are the remaining neurons that participate in signal processing. The neurons on the known side process and generalize the information, activating associative neurons that generate feedback signals to the unknown side. The neurons on the unknown side then pass the feedback signal back to the unknown parts of the input. The missing associated inputs can thus be recovered. This figure shows only a conceptual view of separated known and unknown sides. In the case of missing value recovery (AA), there is no such separation, whereas in HA learning and classification associative neurons may be far apart.

## IV. Associative Learning Networks Configuration and Simulation Results

This section discusses the effect of network topology on the associative learning quality. A classification application was used to illustrate the associative learning results in SOLAR structures. Data presented to SOLAR were binary coded to simulate a biologically plausible operation.

### A. Iris Database Coding and Its Simulation

This associative learning network has been tested on the Iris plants database [24]. The Iris database has three classes, four numeric features, and 150 instances. These classes are defined by three types of Iris plant (Iris Setosa, Iris Versicolour, and Iris Virginica) each containing 50 instances. The numeric features include sepal length, sepal width, petal length, and petal width (all in centimeter). Table III shows the statistics of the database.

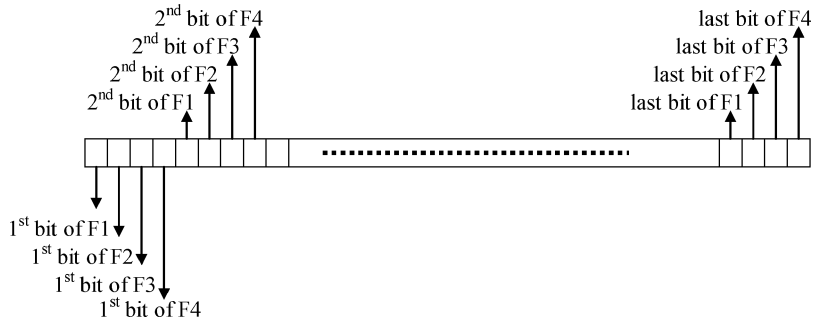



Fig. 5. Merging coded feature values.

TABLE III
IRIS DATABASE STATISTICS

| | Min | Max | Mean | STD | Class correlation |
|---|---|---|---|---|---|
| Sepal length | 4.3 | 7.9 | 5.84 | 0.83 | 0.7826 |
| Sepal width | 2.0 | 4.4 | 3.05 | 0.43 | 0.4194 |
| Petal length | 1.0 | 6.9 | 3.76 | 1.76 | 0.9490 |
| Petal width | 0.1 | 2.5 | 1.20 | 0.76 | 0.9565 |

Since the outside input is expected to be presented in a binary form, each feature was scaled linearly and coded using a sliding bar code with $M$ bits, as shown in Fig. 4. Let $\mathrm{Max}$ and $\mathrm{Min}$ represent the maximum and minimum values of the feature, respectively. To code a scaled feature $V$, we set $L$ consecutive bits from bit $(V-\mathrm{Min})+1$ to $(V-\mathrm{Min})+L$ to 1, which is shown as the shadowed area in Fig. 4. The remaining bits were set to 0, and are shown as the blank area. $L$ has been used to introduce input redundancy, such that a sufficient number of randomly connected neurons could be stimulated by the input signal. All the features were scaled between zero and 30, the value of $L$ for each feature was set to 12 bits, and the total number of input bits per single feature was 42.

The features were combined with the class ID to form the input to the associative network. In order to balance the probability of each feature being associated with the sample class ID, we merged the four features into an input pattern vector, as shown in Fig. 5. The total bit length of the input pattern vector is 168.

Orthogonal binary sequences were used to code class IDs. During testing, feedback from the network was used to generate an output code for each sample. The output code was then compared against each of the class ID codes. The class ID that had the minimal Hamming distance to the output code was taken as the classification result. Ten-fold cross-validation was used for the Iris database.

Fig. 6 shows the association made in the testing of one pattern. The $y$-axis shows the input numbers. The target pattern is presented on inputs 1-168, while the associated class ID pattern is presented on inputs 174–341. The $x$-axis represents the distance from the input (number of layers). All the actively associating neurons are represented by filled circles. One of these actively associating neurons is shown with its feedback path as an example of the association between the target pattern and the class ID pattern. The small squares represent the other neurons on this feedback path, and the stars along the $y$-axis stand for

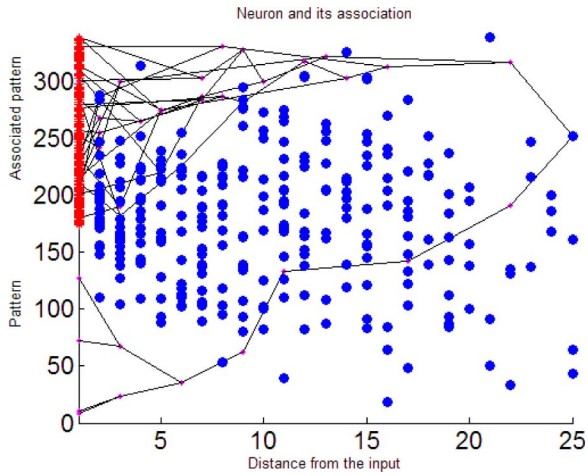



Fig. 6. Actively associating neurons and a selected feedback path in the SOLAR network.

the class ID code bits that were correctly recognized. The remaining associative inputs were either not activated (unknown input values) or recognized incorrectly.

A zoomed-in view is provided in Fig. 7, which shows a subset of neurons near the associated class ID pattern inputs. All the neurons that participated in making associations were illustrated, including neurons in the feedback paths of actively associating neurons. As can be seen in Fig. 6, the actively associating neurons needed to solve this problem are distributed from the first layer to the twenty-fifth layer, which indicates the depth of association.

After ten-fold cross-validation, the overall correct classification rate for Iris database was 86%. The performance comparison with other methods is presented in Section IV-D. This classification rate can be further improved by applying subtle differences in the network structure as discussed in the rest of this section.

### B. Simulation of a Hierarchical Structure

This part investigates how the network topology affects learning ability and the required hardware resources. We are also interested in the relationship between the number of hierarchy levels and problem complexity, the effect of local interconnections and redundancy of information on learning ability, and the statistical robustness of results.

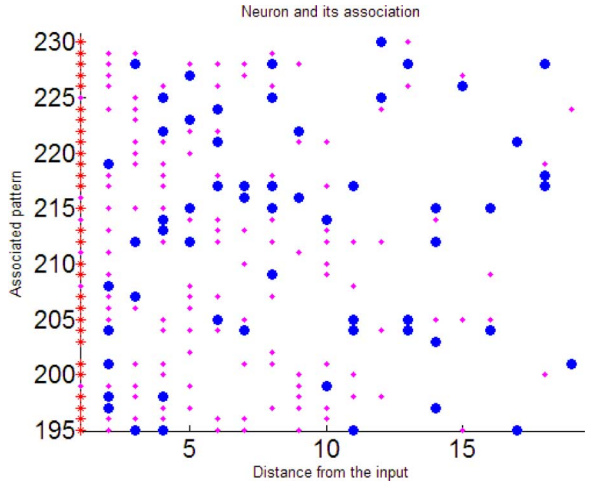



Fig. 7. Class ID inputs correctly recognized by associations, zoomed into Fig. 6.

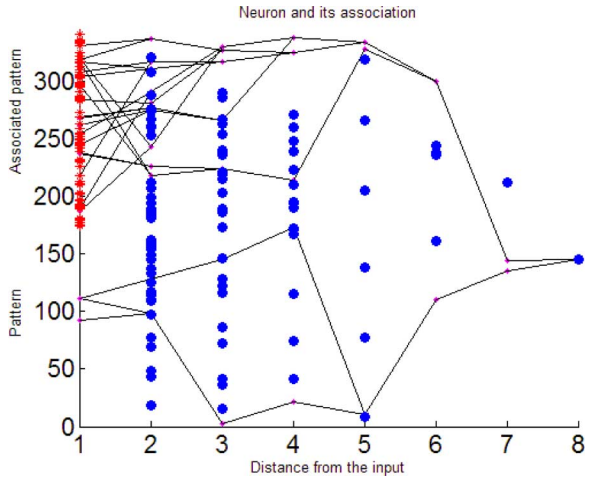



Fig. 8. Actively associating neurons and a selected feedback path in the hierarchical SOLAR network.

In the simulation of the Iris databases in Section IV-A, the neurons were randomly connected to their neighbors with a Gaussian distribution of the distances between neurons, in the vertical direction (rows) and in the horizontal direction (layers). The standard deviations of these distances in both directions were set to 30 and 5, respectively. To observe the influence of the input distribution, we first rebuilt the network using a hierarchical structure, by restricting each neuron to connect only to the neurons of the previous layer. In addition, in the vertical direction, the vertical connection probability distribution was a superposition of a Gaussian distribution and a uniform distribution. Half of vertical connections went to local neurons (using a Gaussian distribution with zero mean and standard deviation equal to two), while the other half of the vertical connections followed the uniform distribution, allowing connections to distant neurons. Such an interconnection scheme delivers the topology of a small world network as defined in [21]. We still use the Iris database for association learning. Using the same symbols as in Fig. 6, Fig. 8 shows the actively associating neurons with an example of feedback path made by the hierarchical network. Fig. 9 is a zoomed-in view near the associated class ID inputs.

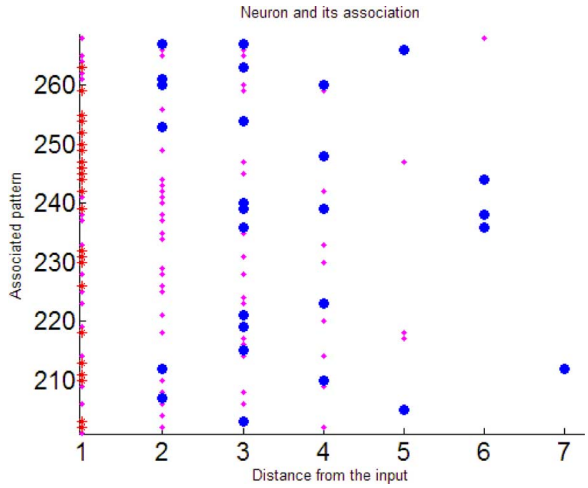



Fig. 9. Class ID inputs correctly recognized by associations, zoomed into Fig. 8.

TABLE IV
CLASSIFICATION RATE VERSUS PERCENTAGE OF VERTICAL CONNECTIONS WITH UNIFORM DISTRIBUTION

| Percentage | 0 | 10 | 20 | 30 | 40 | 50 |
|---|---|---|---|---|---|---|
| Correct classification rate | 12 | 30 | 51.33 | 70 | 72.67 | 83.33 |
| Percentage | 60 | 70 | 80 | 90 | 100 | |
| Correct classification rate | 82.67 | 82.67 | 82 | 84 | 88 | |

After all 150 patterns were trained and tested through the ten-fold cross-validation, we got the overall correct classification rate 86%. Apparently, fewer actively associating neurons were needed and they only spread to the eighth layer, which indicates the depth of association in this network. We conclude that with a hierarchical network structure fewer layers are needed to establish necessary associations with similar correct classification rate.

### C. Simulation of Small World Networks

In this section, we would like to demonstrate how the classification performance varies with different wiring schemes in the small world network structure. As described in Section IV-B, the connection probability distribution in the vertical direction is a superposition of a Gaussian distribution and a uniform distribution. Table IV shows how the correct classification rate changes with the percentage of uniformly distributed local connections.

From the results presented in Table IV, it can be seen that without the superposition of the uniform distribution (0%) neurons cannot connect to distant neighbors, and, therefore, they may not be able to create sufficient associations. As a result, the correct classification rate in this case is the lowest. It is also found that as the percentage of vertical connections following uniform distribution increases from 0% to 50%, the correct classification rate increases. However, beyond 50%, the correct classification rate remains stable within 82% and 88%.

An alternate way of increasing distant neuron connections is to use a larger standard deviation of the Gaussian distribution. Table V shows the correct classification rates obtained by

TABLE V
CORRECT CLASSIFICATION RATE VERSUS GAUSSIAN STANDARD DEVIATION IN VERTICAL DIRECTION

| Standard deviation | 5 | 15 | 25 | 35 | 50 |
|---|---|---|---|---|---|
| Correct classification rate | 51.33 | 54.67 | 60.67 | 70 | 88 |

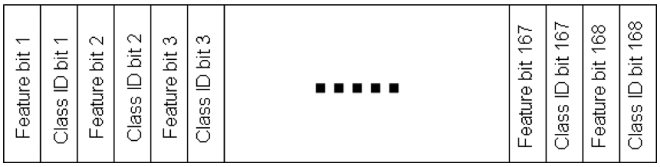



Fig. 10. Mixed features and class ID code.

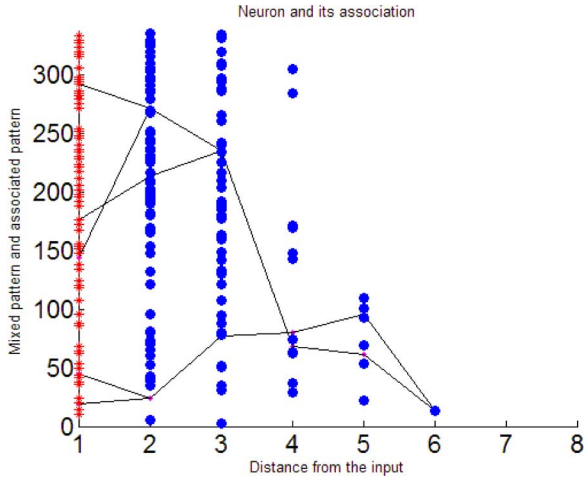



Fig. 11. Actively associating neurons and a selected feedback path in the hierarchical SOLAR network with mixed feature and class ID inputs.

fixing the percentage of uniformly distributed vertical connections to 20% and varying the standard deviation of the Gaussian distribution.

As can be seen from Table V, the correct classification rate increases from 51.33% to 88% as the standard deviation of the Gaussian distribution increases from 5 to 50.

In conclusion, the analysis of this part indicates that one of the most critical factors in the wiring scheme is distant connectivity for a classification task.

### D. Mixing Features and Class ID Code

Instead of changing distribution parameters to improve distant connectivity, we can simply mix the merged feature code shown in Fig. 5 with the class ID code, as shown in Fig. 10. Such a mixed input pattern vector automatically guarantees sufficient interconnection between the features and the class ID.

Again, the Iris database was used. Similar to Fig. 6, Fig. 11 shows the actively associating neurons with a feedback path made by this hierarchical SOLAR network with mixed inputs. Fig. 12 is a zoomed-in view near some mixed inputs. As expected, the depth of associations is even less in this network than the one shown in Fig. 8. An overall correct classification rate of 94.67% was obtained after the ten-fold cross-validation.

In order to further improve the classification performance and the stability of the results, multiple networks were

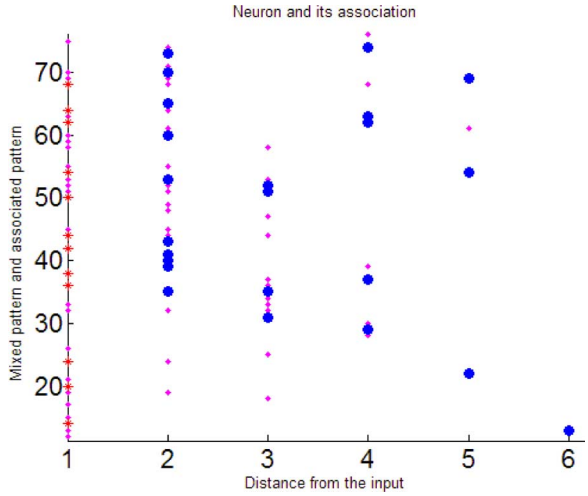



Fig. 12. Class ID inputs correctly recognized by associations, zoomed into Fig. 11.

used in parallel on the same target problem. Based on the Hamming distance between the output and the class ID code, a winner-takes-all voting scheme was used to combine their recognition outputs. The overall correct classification rate reached 96.67% with five parallel networks. For comparison, results reported in [25] give correct classification rates for the Iris database between 91.33%–97.33%. Obviously, mixing the features and class ID together with the winner-takes-all voting scheme results in better classification and fewer layers.

### E. Problem Complexity and Hierarchical Depth

We selected the Glass Identification Database [26] to study the effect of problem complexity on the required depth of associative network processing. This database contains 214 instances, nine features, and six classes. The features selected in this database are as follows:

1) RI: refractive index;
2) Na: sodium (unit measurement: weight percent in corresponding oxide, as are attributes 3–9);
3) Mg: magnesium;
4) Al: aluminum;
5) Si: silicon;
6) K: potassium;
7) Ca: calcium;
8) Ba: barium;
9) Fe: iron;

and the six classes of different glass contain: building_windows_float_processed, building_windows_non_float_processed, vehicle_windows_float_processed, containers, tableware, and headlamps (terminology used in description of Glass database [26]).

We first trained and tested the network with all instances from all six classes. In order to evaluate the effect of the problem complexity on the depth of association, we also tested the network on half of the database containing only three classes. Fig. 13 shows the average number of actively associating neurons at each layer in base-2 logarithmic scale.

When the network is dealing with six classes, there was substantial amount of active associative neurons even in the twenty-

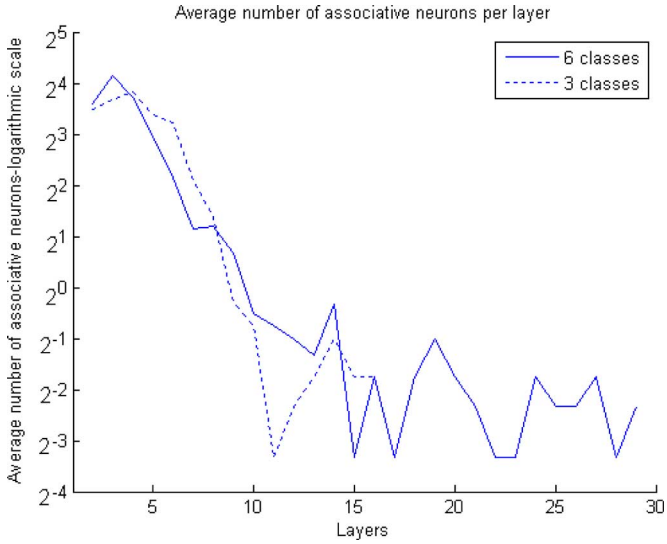



Fig. 13. Average number of actively associating neurons per layer.

ninth layer, whereas such neurons were only found within the sixteenth layer when dealing with three classes. The depth of association differs by 13 due to higher problem complexity. Neurons located high in the hierarchy (i.e., more distant from the input) are believed to have better generalization and comprehension on input information. Thus, more complex problems may require more layers, and in turn use more resource of the associative network.

Although only two classification applications have been presented in this part, it should be noticed that SOLAR was not designed to be a classifier. As can be seen from Section IV-A, the features and class IDs were coded in a similar way and presented to the network together. Since the network itself does not differentiate pattern or classification code, it is not limited to such HA applications. The network will also be suitable for AA learning tasks by nature. Its learning capacity will be discussed in future work.

## V. Conclusion

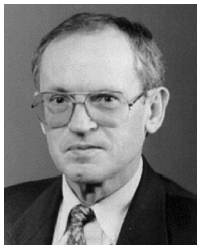
This paper presents an associative learning network based on a hierarchical SOLAR structure. SOLAR is a biologically inspired machine-learning concept. It is a sparsely connected network organized as a fixed lattice of distributed, parallel processing units (neurons). Compared with ANNs, it has a flexible interconnect structure designed to emulate the organization of the human cortex structure that is both plausible for a biological network and easy to implement in hardware. The associative learning SOLAR network described in this paper is constructed as a fixed interconnection network with feedback and inhibitory links. Similar to Vogel's distributed AA memories [7], SOLAR learns to associate patterns that can be defined on a subset of input neurons. SOLAR discovers the correlation between inputs and establishes associations inside the neurons, without a need to differentiate between the associated class ID patterns and target data patterns. It is capable of handling a wide variety of machine learning tasks including image recognition, classification, and data recovery, and is suitable for online learning. This work validates the associative learning ability of the SOLAR structure.

The current realization is straightforward and rather primitive. SOLAR neurons execute very simple functions and their firing behavior is not as natural as biological neurons. The SOLAR organization will be further modified to develop an advanced machine intelligence system capable of associative learning, adaptation, and value driven interaction with the environment.

## References

[1] D. E. Rumelhart and J. L. McClelland, *Parallel Distributed Processing*. Cambridge, MA: MIT Press, 1986, PDP Research Group.
[2] R. Hecht-Nielsen, *Neurocomputing*. Reading, MA: Addison-Wesley, 1990.
[3] B. Kosko, "Bidirectional associative memories," *IEEE Trans. Syst., Man, Cybern.*, vol. 18, no. 1, pp. 49–60, Jan./Feb. 1988.
[4] L. Wang, "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 1, pp. 73–82, Feb. 1999.
[5] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," in *Proc. Nat. Acad. Sci.*, Apr. 1982, vol. 79, pp. 2554–2558.
[6] D. Vogel and W. Boos, "Sparsely connected, Hebbian networks with strikingly large storage capacities," *Neural Netw.*, vol. 4, no. 10, pp. 671–682, 1997.
[7] D. Vogel, "Auto-associative memory produced by disinhibition in a sparsely connected network," *Neural Netw.*, vol. 5, no. 11, pp. 897–908, 1998.
[8] D. Shen and J. Cruz, Jr., "Encoding strategy for maximum noise tolerance bidirectional associative memory," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 293–300, Mar. 2005.
[9] S. Du, Z. Chen, Z. Yuan, and X. Zhang, "Sensitivity to noise in bidirectional associative mMemory (BAM)," *IEEE Trans. Neural Netw.*, vol. 16, no. 4, pp. 887–898, Jul. 2005.
[10] S. Arik, "Global asymptotic stability analysis of bidirectional associative memory neural networks with time delays," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 580–586, Mar. 2005.
[11] S. Chartier and M. Boukadoum, "A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 59–68, Jan. 2006.
[12] ——, "A bidirectional heteroassociative memory for binary and grey-level patterns," *IEEE Trans. Neural Netw.*, vol. 17, no. 2, pp. 385–396, Mar. 2006.
[13] L. Wang, "Complex-valued multistate neural associative memory," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1491–1496, Nov. 1996.
[14] J. Buhmann and K. Schulten, "Noise-driven temporal association in neural networks," *Europhys. Lett.*, vol. 4, pp. 1205–1209, 1987.
[15] D. O. Hebb, *The Organization of Behavior*. New York: Wiley, 1949, p. 44.
[16] I. Guyon, L. Personnaz, J. P. Nadal, and G. Dreyfus, "Storage and retrieval of complex sequences in neural networks," *Phys. Rev. A*, vol. 38, pp. 6365–6372, 1988.
[17] D. L. Wang and B. Yuwono, "Anticipation-based temporal pattern generation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 4, pp. 615–628, Apr. 1995.
[18] L. Wang, "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 29, no. 1, pp. 73–82, Feb. 1999.
[19] J. A. Starzyk, Z. Zhu, and T.-H. Liu, "Self-organizing learning array," *IEEE Trans. Neural Netw.*, vol. 16, no. 2, pp. 355–363, Mar. 2005.
[20] M. S. Keshavan and G. E. Hogarty, "Brain maturational processes and delayed onset in Schizophrenia," *Development and Psychopathology*, vol. 11, no. 3, pp. 525–543, 1999.
[21] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, pp. 440–442, Jun. 4, 1998.
[22] W. T. Greenough, J. Black, and C. Wallace, "Experience and brain development," *Child Development*, vol. 58, pp. 539–559, 1987.
[23] J. A. Starzyk, M. Ding, and H. He, "Optimized interconnections in probabilistic self-organizing learning," in *Proc. Int. Conf. Artif. Intell. Appl. (IASTED )*, Innsbruck, Austria, Feb. 14–16, 2005, pp. 479–484.

[24] R. A. Fisher, Iris Plants Database. (July 1988) [Online]. Available: http://faculty.cs.byu.edu/~cgc/Teaching/CS_478/iris.arff
[25] R. Andonie and A. Cataron, "An informational energy lVQ approach for feature ranking," in *Proc. Euro. Symp. Artificial Neural Netw. (ESANN)*, M. Verleysen, Ed., Bruges, Belgium, Apr. 28–30, 2004, pp. 471–476.
[26] B. German, Glass Identification Database. (Sep. 1987) [Online]. Available: http://www.ailab.si/orange/doc/datasets/glass.htm



**Janusz A. Starzyk** (SM'83) received the M.S. degree in applied mathematics and the Ph.D. degree in electrical engineering from Warsaw University of Technology, Warsaw, Poland, in 1971 and 1976, respectively.

From 1977 to 1981, he was an Assistant Professor at the Institute of Electronics Fundamentals, Warsaw University of Technology. From 1981 to 1983, he was a Postdoctorate Fellow and Research Engineer at McMaster University, Hamilton, ON, Canada. In 1983, he joined the Department of Electrical and Computer Engineering, Ohio University, Athens, where he is currently a Professor of electrical engineering and computer science. He has cooperated with the National Institute of Standards and Technology in the area of testing and mixed signal fault diagnosis. He has been a consultant for AT&T Bell Laboratories, Sarnoff Research, Sverdrup Technology, Magnolia Broadband, and Magnetek Corporation. His current research is in the areas of self-organizing learning machines, neural networks, rough sets, VLSI design and test of mixed signal complementary-metal–oxide–semiconductor (CMOS) circuits, and reconfigurable design for wireless communication.

**Zhen Zhu** (S'06) received the B.S. degree in information and communication engineering from Xi'an Jiaotong University, Xi'an, China, in 2000 and the M.S. degree in electrical engineering from Ohio University, Athens, in 2002, where currently, he is working towards the Ph.D. degree.

He has been working as a Research Associate at the Avionic Engineering Center, Ohio University, since 2001. He also worked for FreeFlight Systems, Waco, TX, in 2004. His research interests include machine learning systems, global positioning system receiver design, and signal processing.

**Yue Li** received the B.S. degree in communication engineering from Nanjing University of Science and Technology, Nanjing, China, in 2003 and the M.S. degree in electrical engineering from Ohio University, Athens, in 2006.

In 2006, she joined O2Micro, Inc., Santa Clara, CA, where she is responsible for patent marketing and patent applications in the company.