Complex Temporal Sequence Learning Based on Short-term Memory

DELIANG WANG AND MICHAEL A. ARBIB

We design neural networks to learn, recognize, and reproduce complex temporal sequence, with short-term memory (STM) modeled by units comprising recurrent excitatory connections between two neurons (a dual neuron model). The output of a neuron has graded values instead of binary ones. By applying the Hebbian learning rule at each synapse and a normalization rule among all synaptic weights of a neuron, we show that a certain quantity, called the input potential, increases monotonically with sequence presentation, and that the neuron can only be fired when its input signals are arranged in a specific sequence. These sequence-detecting neurons form the basis for our model of complex sequence recognition, which can tolerate distortions of the learned sequences. A recurrent network of two layers is provided for reproducing complex sequences.

I. Introduction

The ability to understand one's environment, essential for intelligence, is not static. The order in which events occur can be even more important than the events themselves, and an intelligent system, whether it be a frog, a robot, or a human, must be able to detect this ordering and to reproduce this ordering on some *cue*. Yet many attempts to model neural networks, such as associative memory [1] and the Boltzmann machine [2], dealt only with static equilibrium rather than with ordering of patterns.

Generally, a *temporal sequence* $\mathcal{S}$ is defined as: $p_1 - p_2 - \cdots - p_m$. Each $p_i (i = 1, \cdots, m)$ is called a component of $\mathcal{S}$ (sometimes we call it a *spatial pattern*, or just a *symbol*). The *length* of a sequence is the number of components in the sequence. In general, a sequence may include repetitions of the same subsequence in different contexts. For example, $\mathcal{S}_1$: *C-A-B-D-A-B-E* contains repetitions of subsequence *A-B*, and such a subsequence is called a *recurring subsequence*. The correct successor can be determined only by knowing symbols prior to the current one. We refer to the prior subsequence required to reproduce the current symbol p_i in $\mathcal{S}$ as the *context* of p_i and the length of this prior subsequence as the *degree* of p_i. The symbol *D* in $\mathcal{S}_1$, for example, has a degree of 3. The *degree of a sequence* is defined as the maximum degree of its components. A 1 degree sequence is called *a simple sequence*, and otherwise a sequence is *a complex sequence*. If a recurring subsequence of $\mathcal{S}$ contains in itself another recurring subsequence, e.g. *A-B-A* in *A-B-A-C-A-B-A-D*, $\mathcal{S}$ is called a *high-order complex sequence*, otherwise a *first-order complex sequence*.

Neural networks to store and recognize a temporal sequence of input stimuli have been previously studied. Grossberg [3] demonstrated one neural network called the *outstar avalanche* that can be used to generate temporal patterns. The outstar avalanche is composed of n sequential outstars. Any outstar $\mathfrak{M}_i$ can store a spatial pattern and be activated by a signal in the vertex v_i. These vertices are connected as $v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_n$, and a signal from v_i arrives with some delay at v_{i+1}. So an initial signal at v_1 can produce sequentially the spatial patterns stored in $\mathfrak{M}_1, \mathfrak{M}_2, \cdots, \mathfrak{M}_n$ respectively. Based on the anatomy of the dentate gyrus region of the mammalian hippocampus, Stanley and Kilmer [4] designed a network called the *wave model* which can learn sequences of inputs separated by certain time intervals and reproduce these sequences when cued by their initial subsequences. Recently, using a bidirectional associative memory built from two fields of fully connected neurons, Kosko [5] showed that by feeding the spatial pattern output from one field back to the other field, the network can generate a sequence of patterns over time that alternates between the two fields.

Using a synaptic triad made up of three neurons *A-B-C* as building blocks, Dehaene *et al.* [6] proposed a layered neural network, called the *selection model*, which can recognize temporal sequences. The description of a synaptic triad guarantees that neuron *B* is activated only when *A* and *C* appear in the order *C-A*, that is, neuron *B* is made sequence-sensitive. A network of synaptic triads can be constructed for a sequence of any length, which may include some repetitions of a part of the sequence. In order to make the neural network able to learn an arbitrary sequence, connections among these synaptic triads are made randomly and the resulting network can be selected by an input sequence. This selection model is based on the *ad hoc* assumption on the architecture of the network, so learning is severely limited by the immense connections

Manuscript received September 14, 1989; revised March 16, 1990. The research described in this paper was supported in part by grant no. 1RO1 NS 24926 from the National Institutes of Health (M. A. Arbib, Principal Investigator).

D. Wang is a student at the University of Southern California, Los Angeles, CA 90089, USA.

M. A. Arbib is with the Center for Neural Engineering, University of Southern California, Los Angeles, CA 90089-2520, USA.

IEEE Log Number 9038829.

0018-9219/90/0800-1536$01.00 © 1990 IEEE

that would be required to learn an arbitrary temporal sequence which is not trivially short.

Storage of temporal sequences in the spin-like Hopfield network has been proposed recently by several authors (see [7]-[12]). In this paradigm, each pattern is stable over some time period, at the end of which a sharp transition leading to the next pattern occurs due to stored transitions between consecutive patterns. One difficulty is the storage and retrieval of complex sequences. In most of these models, a given pattern can occur only once among all the stored sequences, which is a severe restriction.

In this paper, we propose a different approach for storage of temporal sequences. First of all, a dual neuron model is used for storing a signal for a short time. The output of this gradually decaying dual neuron is a graded signal, rather than the binary signal used in many neural network models. Following the Hebbian training [13] of ordered graded signals a neuronal quantity, *the input potential*, which is the weighted sum of the ordered inputs, is shown to increase monotonically until it saturates. After this training, if we set the threshold of the neuron to the saturation point of its input potential, then this neuron can only be activated by this specific sequence of inputs. This property naturally leads to the concept of a *sequence-detecting neuron*. An important thing is that, after learning, this type of neuron is fired by a previous *sequence* of patterns, not just a previous pattern, so it overcomes the limitation of networks which can only generate simple sequences. The same idea is used for recognizing any complex sequence. Furthermore, we show that by adding another sequence-detecting layer, any complex sequence can be reproduced.

II. Temporal Sequence Recognition

A. Dual Neuron Model of STM

In order to link two temporally discontiguous patterns, the previous pattern has to be preserved for a certain period of time. This temporal link can be provided by *short-term memory* (STM). STM has been extensively studied, and is suggested to be physiologically due to recurrent excitatory connections. This physiological explanation is adopted in many efforts of neural network modeling (for example, see [14], [15]). To simplify the process while preserving the basic idea, we use a dual neuron (Fig. 1(a)) to model STM. The two neurons N_1 and N_2 have activities or membrane potentials $m_1(t)$ and $m_2(t)$ described using the leaky integrator model (in discrete form)

$$\begin{cases} m_1(t + \Delta t) = m_1(t) + \Delta t[-Km_1(t) \\ \qquad\qquad + T_{12}m_2(t + \Delta t - \tau/2) + I(t)] \\ m_2(t + \Delta t) = m_2(t) + \Delta t[-Km_2(t) \\ \qquad\qquad + T_{21}m_1(t + \Delta t - \tau/2)] \end{cases} \tag{1}$$

where K is a relaxation constant, T_{12} and T_{21} are synaptic weights, Δt is a discretization interval, and τ is the cycle time for a signal to travel between the two neurons. $I(t)$ represents external input to this dual neuron. Equation (1) can form a single damped oscillatory system with period τ by choosing appropriate parameters. Although similar in form to other coupled oscillators (for example, see [16]), the oscillation created here damps. Each time the signal appears on

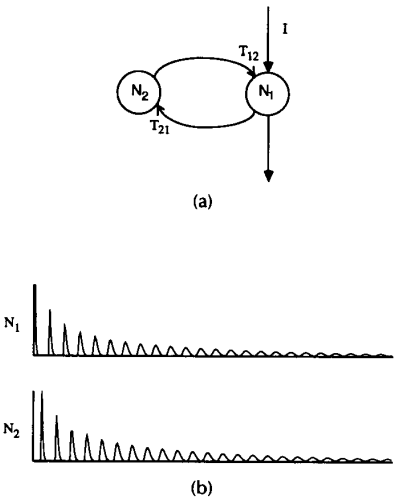


Fig. 1. a) A dual neuron. b) Response of a dual neuron model, which maintains a signal for a certain memory span. The model parameters are: $K = 8.3$, $T_{12} = 6.5$, $T_{21} = 10.0$, $\tau = 12\Delta t$, $\Delta t = 0.1$.

N_1, the amplitude of the signal decreases. This decrease will be designated by function $g(l\tau)$, which is monotonically decreasing ranging between 1 and 0 with $g(0)$ equal to 1. Analysis shows that $g(l\tau)$ is an exponential decay function, as manifested in Fig. 2(b) which shows the response of a dual neuron model when N_1 is stimulated initially. This represents a basic model for STM.

B. Sequence-Detecting Neuron Model

A biological neuron updates its state (firing and silent) in a few milliseconds, a typical presentation of a symbol in a sequence lasts several hundred milliseconds, and the length of STM is usually in seconds. For simplicity, we will use a single neuron to represent a symbol in a sequence[1], so we use two time scales to model the fact that symbol transition in a sequence is much slower than neuronal state transition. One time scale is for the interaction among symbols (called *symbol scale*), here represented by dual neurons; another is for interaction among neurons within each oscillator (called *neuron scale*). Again to simplify the situation, we choose one step on the symbol scale, denoted as Δ, to equal r rhythmic periods of a dual neuron (namely $\Delta = r\tau$). From the previous section we see that STM can last many Δs, hence many symbol presentations.

The idea for this sequence-detecting neuron model is that sequence learning polarizes the synaptic weights of the detecting neuron in such a way that these polarized weights can form the maximal membrane potential when the learned sequence is presented (*maximization principle*). Suppose that we have n dual neurons $\langle N_{11}, N_{21}\rangle, \langle N_{12}, N_{22}\rangle, \cdots, \langle N_{1n}, N_{2n}\rangle$, where each N_{1i} connects to all the neurons $N_{1j} (i \neq j)$ in the network. The activity of neurons N_{1i} and N_{2i} are $m_{1i}(t)$ and $m_{2i}(t)$ respectively, and are defined as (compare Eq. (1)):

[1]One single neuron here should be viewed biologically as a neuron assembly.

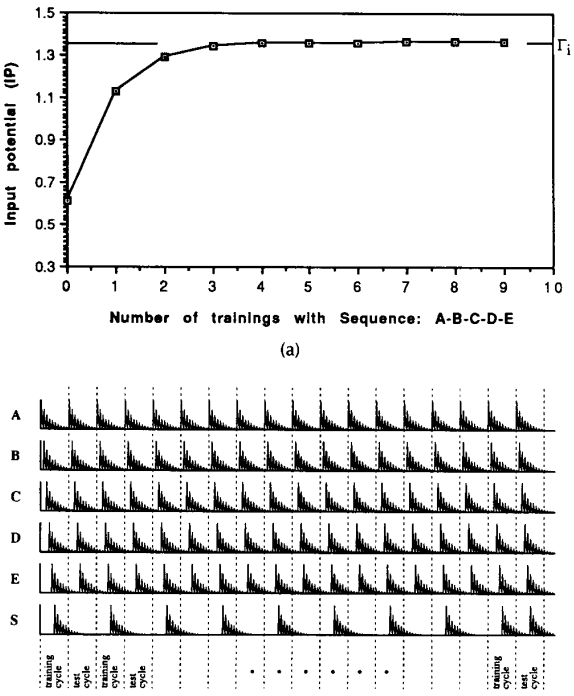


Fig. 2. a) Input potential increases monotonically with number of training trials. b) Training for sequence recognition. Ten dual neurons have been modeled, and temporal activities of N_{11}, N_{12}, $\cdots$, N_{15}, and $N_{1,10}$ are displayed in the figure from top to bottom. Note that in each case, a dual neuron must fire to first achieve a nonzero membrane potential, but thereafter activity decays according to the curve g (as shown in Fig. 1(b) on an expanded time scale). A symbol indicates which pattern the corresponding neuron represents. The sequence to be detected is $\mathbb{S}_2$: *A-B-C-D-E*. During each training cycle, the sequence is presented, followed by an activation of a sequence detecting neuron *S*. Each training cycle is followed by a test cycle, during which the sequence is presented alone, i.e. without a following activation of *S*, in order to see if neuron *S* can be activated by the sequence. After 9 trainings the sequence detecting neuron *S* can be activated by another presentation of the sequence. The parameters are: $M = 20$, $B = 2.0$, $C_i = 0.4$, $T_{12} = 6.5$, $T_{21} = 10.0$, $\Gamma_{10} = 1.3634$, $K_{1i} = K_{2i} = 5.0$, $\tau = 6\Delta t$, $\Delta = \tau$, $\Delta t = 0.1$.

$$\hat{m}_{1i}(t + \Delta t) = \begin{cases} m_{1i}(t) + \Delta t\Big[-K_{1i}m_{1i}(t) + T_{12}m_{2i}(t + \Delta t - \tau/2) \\ \quad + Mf\Big(\sum_{j \neq i} W_{ij}m_{1j}(t + \Delta t - \Delta) + I_i(t) - \Gamma_i\Big)\Big], \\ \quad \text{if } t \bmod \Delta = 0 \\ m_{1i}(t) + \Delta t[-K_{1i}m_{1i}(t) + T_{12}m_{2i}(t + \Delta t - \tau/2)], \\ \quad \text{otherwise} \end{cases} \tag{2}$$

$$m_{1i}(t + \Delta t) = \text{Min}\,(\hat{m}_{1i}(t + \Delta t), B) \tag{3}$$

$$f(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

$$m_{2i}(t + \Delta t) = m_{2i}(t) + \Delta t[-K_{2i}m_{2i}(t) + T_{21}m_{1i}(t + \Delta t - \tau/2)] \tag{5}$$

In (2) Γ_i is the threshold for N_{1i} for input other than from N_{2i}. $I_i(t)$ represents external input to N_{1i}. The condition in (2) that $t \bmod \Delta = 0$ describes the time scale for the presentation of symbols. In real situations, symbols are presented continuously, and this is discretized by Δ in our model. The corresponding $m_{1i}(t)$ at these time instants ($t \bmod \Delta = 0$) can be interpreted as the average values over time interval Δ, or as the value at the end of each presentation of a symbol. W_{ij} is the synaptic weight from neuron N_{1j} to N_{1i}, and the summation in (2) represents interactions among dual neurons. M is a gain factor. Equation (3) stipulates a maximal activity B for N_{1i}. From now on, we say neuron N_{1i} is firing at time t if $m_{1i}(t) = B$. The formulation for N_{2i} is similar to that for N_{1i} except that N_{2i} only receives input from N_{1i}. T_{12} and T_{21} are fixed weights between N_{2i} to N_{1i}, and they are the same for all n dual neurons. Activities $m_{1i}(t)$ and $m_{2i}(t)$ have graded values with maximum B. Once N_{1i} fires, this signal will oscillate between the dual neuron N_{1i} and N_{2i} with damping until the signal totally vanishes, if no further input can activate the N_{1i}. Note that the formulas are such that nonfiring neurons can still affect the state of other neurons.

Synaptic learning follows the Hebbian learning rule [13] for modification and a later normalization [17]. Again, learning takes place on the symbol scale (Δ) since only the weights of connections among oscillators are changed.

$$\begin{cases} \hat{W}_{ij}(t) = W_{ij}(t - \Delta) + C_i f[m_{1i}(t) - B]m_{1j}(t) \\ W_{ij}(t) = \hat{W}_{ij}(t) \Big/ \sum_{j' \neq i} \hat{W}_{ij}(t) \end{cases} \tag{6}$$

where C_i is a gain factor of learning. Note that $f[m_{1i}(t) - B]$ is 0 unless N_{1i} fires. In general, the larger is C_i, the faster is learning and the more easily is the memory value overwritten by a new stimulus, so the choice of C_i reflects a balance between learning speed and stability. According to (6), the effect of learning on any neuron is to change the distribution of all weights to that neuron, so it is reasonable to assume that initially $W_{ij} = 1/(n - 1)$.

In the remaining part, our discussion will be based on a "grandmother cell" representation in which each neuron N_{1i} represents either one spatial pattern or a sequence detector to simplify the understanding of sequence recognition, as in [6], [9]. Suppose that we train neuron N_{1i} to detect a sequence $\mathbb{S}_i$. This is done in our model by activating N_{1i} (setting $I_i(t)$ at a high level) immediately after the presentation of $\mathbb{S}_i$. Therefore this is a kind of supervised learning. As we will see later, this kind of training is very natural for sequence reproduction. We define $\mathbb{S}_i = p_{i_1} - p_{i_2} - \cdots - p_{i_k}$, $1 \leq i_j \leq n$ and $i_j \neq i$ ($j = 1, \cdots, k$), where pattern p_{i_j} fires (is represented by) neuron N_{1i_j}. In this section we only consider $\mathbb{S}_i$ as a simple sequence. Since we are now only concerned about recognizing $\mathbb{S}_i$, we can simplify the original fully connected network into the one where N_{1i} is projected upon by N_{1i_j} ($j = 1, \cdots, k$) and all other connections among dual neurons are left out.

Let us define the input potential IP_i of $\mathbb{S}_i$ to N_{1i} (since dual neuron i is only for $\mathbb{S}_i$) as the weighted sum to N_{1i} immediately after the presentation of $\mathbb{S}_i$ (we use t' to indicate these

time instants)

$$IP_i = \sum_{j=1}^{k} W_{ii_j} m_{1i_j}(t') = B \sum_{j=1}^{k} W_{ii_j} g((k-j)\Delta) \tag{7}$$

where $g(l\Delta)$ is as introduced earlier (Δ is a multiple of τ). This formula follows because when pattern p_{ij} is presented, N_{1i_j} is activated and $m_{1i_j}(t)$ reaches value B. Equations (2) and (3) together guarantee that once N_{1i_j} is activated due to either an external input or a summed input from other neurons, its activity drops down monotonically on the symbol time scale if further input to N_{1i_j} cannot fire it. In this situation, indeed, further input to N_{1i_j} cannot fire it, because (i) since $\mathbb{S}_i$ is a simple sequence, N_{1i_j} can only be activated externally once by pattern p_{i_j} during the presentation of $\mathbb{S}_i$; and (ii) besides the presentation time of p_{i_j}, the summed input from other neurons $N_{1i}(l \neq j)$ can never activate N_{1i_j} since this summed input cannot overcome the threshold of N_{1i_j}, which can be chosen as a parameter. At the completion of the presentation of the entire sequence, $m_{1i_j}(t)$ drops to $B \cdot g((k-j)\Delta)$. The external input to a dual neuron can only affect it through the binary gate $f(x)$ in (2), and the dual neuron oscillates at its own pace if nothing is further gated in. The precise form of $g(l\Delta)$ is not important, and the only thing that matters for our later analysis is that $g(l\Delta)$ is monotonically decreasing, which is satisfied in our model.

Theorem 1: Repeated training with only $\mathbb{S}_i$ results in all weights to N_{1i} following the distribution: $W_{ii_j} = Cg((k-j)\Delta)(j = 1, \cdots, k)$, and all $W_{il} = 0$ $(l \neq i_1, \cdots, i_k)$, where C is a constant.

Proof. According to (6), the synaptic weights to N_{1i} only change when N_{1i} is firing, which is immediately after a presentation of $\mathbb{S}_i$ due to the supervised way of learning. Let us denote $W^0_{ii_j}$ as W_{ii_j} before any training, and $W^m_{ii_j}$ as W_{ii_j} after the mth presentation of $\mathbb{S}_i$. It is easily verified, that after the mth presentation of $\mathbb{S}_i$, following (6) we have:

$$W^m_{ii_j} = \frac{W^0_{ii_j}}{Q^m} + C_i B g((k-j)\Delta) \sum_{l=1}^{m} \frac{1}{Q^l} \tag{8}$$

where $Q = 1 + C_i B \sum_{j'=1}^{k} g((k-j')\Delta)$. Since $Q > 1$, we get $\lim_{m\to\infty} W^m_{ii_j} = Cg((k-j)\Delta)$ where $C = C_i B/(Q-1)$. For all W_{il}, $l \neq i_1, \cdots, i_k$, we have $W^m_{il} = W^0_{ii_j}/Q^m$ and $\lim_{m\to\infty} W^m_{il} = 0$. Q.E.D.

Here we clearly see that the larger C_i is, the larger Q is, and the more rapidly the weight distribution converges.

Corollary 1. Repeated training with $\mathbb{S}_i$ results in

$$IP_i = BC \sum_{j=1}^{k} g^2((k-j)\Delta) \tag{9}$$

Note that IP_i depends only on the length of sequence $\mathbb{S}_i$.

Theorem 2. During repeated training with $\mathbb{S}_i$, define

$$\begin{aligned}\Delta IP_i^m &= (IP_i \text{ after the } m\text{th presentation of } \mathbb{S}_i) \\ &\quad - (IP_i \text{ after the } (m-1)\text{th presentation of } \mathbb{S}_i) \\ &= B \sum_{j=1}^{k} (W^m_{ii_j} - W^{m-1}_{ii_j})\, g((k-j)\Delta)\end{aligned}$$

Then we have

$$\Delta IP_i^m = \frac{\Delta IP_i^1}{Q^{m-1}} \tag{10}$$

Given (8), the proof is straightforward. Theorem 2 tells us that if the first training with $\mathbb{S}_i$ increases (or decreases) IP_1, then the input potential keeps increasing (or decreasing) in subsequent training.

Corollary 2. If repeated training with $\mathbb{S}_i$ begins from the initial state, that is, $W_{ij} = 1/(n-1)$, then after each training $\Delta IP_i > 0$, which means IP_i increases monotonically with sequence training.

Corollary 1 plus Corollary 2 gives us the insight to build a model for temporal sequence learning. If we choose Γ_i in (2) as the input potential expressed in (9), that is, $\Gamma_i = BC\Sigma_{j=1}^{k} g^2((k-j)\Delta)$ (within a certain small error ϵ), then the result of training with $\mathbb{S}_i$ is to build IP_i in order to fire N_{1i} by the presentation of $\mathbb{S}_i$ alone. In other words, after a certain number of training trials, a presentation of $\mathbb{S}_i$ alone will fire N_{1i}, and neuron N_{1i} will recognize sequence $\mathbb{S}_i$. We say that neuron N_{1i} *has learned* the sequence $\mathbb{S}_i$ if the presentation of $\mathbb{S}_i$ can activate this neuron. Value Γ_i can be set up during the first training, since from (7) and (9) we have $\Gamma_i = BC\Sigma_{j=1}^{k} g^2((k-j)\Delta) = C/B\ \Sigma_{j=1}^{k} m^2_{1i_j}(t')$, namely we can avoid using $g((k-j)\Delta)$ by looking at the corresponding membrane potentials at t'. Conversely, note the interesting fact that Γ_i can be set purely on the basis of the *length* of the sequence. Figure 2 shows a computer simulation of the sequence-detecting neuron model. The curve in Fig. 2(a) reflects the increase of IP_i of a sequence-detecting neuron with number of trainings of sequence $\mathbb{S}_2$: *A-B-C-D-E*. Value Γ_i is set by the system to 1.3634 (where ϵ is chosen as 0.001) and after the ninth training IP_i goes above this threshold, and so the following presentation of the sequence alone activates N_{1i} without the training input I_i. Fig. 2(b) shows the corresponding simulation.

The following theorem guarantees that after a sequence is learned by N_{1i}, *only* the learned sequence can activate N_{1i}, that is, N_{1i} makes no mistake in recognizing the learned sequence.

Theorem 3. After neuron N_{1i} has learned sequence $\mathbb{S}_i$, the presentation of $\mathbb{S}_i$ induces the maximal postsynaptic potential.

Proof outline: Since only one symbol is presented at a time, the nonzero inputs to N_{1i} from the other neurons will at time t' form at most a permutation of $B, Bg(\Delta), \cdots, Bg((k-1)\Delta)$. But by Theorem 1, $W_{ii_j} = Cg((k-j)\Delta)(j = 1, \cdots, k)$ while $W_{il} = 0$ for other l. Thus, by the Cauchy-Schwartz inequality, the maximum input potential is achieved only for the learned sequence.

This theorem realizes the maximization principle that repeated training of a sequence polarizes the weights of the corresponding detecting neuron so that it can only be triggered by this specific sequence. Unlike many other models, this model genuinely stores sequences rather than transition dyads.

C. Recognition of Complex Temporal Sequences

In the previous sections, we proposed the dual neuron model to implement STM, and the property which is used from the STM model is basically an exponential decay membrane potential within the STM period. To concentrate on solving the problem of complex sequence learning, in the following we explicitly incorporate an exponential decay into the model to simulate STM. We thus replace the dual neuron $\langle N_{1i}, N_{2i} \rangle$ by a single neuron N_i. The corresponding neural implementation by dual neurons can be done in the same way as in the previous section.

There is a problem with the above sequence-detecting neuron model if we apply it to arbitrary sequence detection. When a complex sequence, like $\mathcal{S}_3$: *A-B-A-C-A-B-E-B-D*, is presented to the previous model described in (2)–(6), then the later presentation of a recurring pattern will overwrite the signal of the previous presentation maintained in STM. That is, the sequence-detecting neuron can only detect the last presence of a recurring pattern. To solve the overwriting problem, we introduce multiple synapses between two neurons, each of which corresponds to one occurrence within the temporal summation period. The idea is that we replace neuron N_j by an expanded network, as shown in Fig. 3(a), whose terminal P_{jr} remembers the trace for the rth

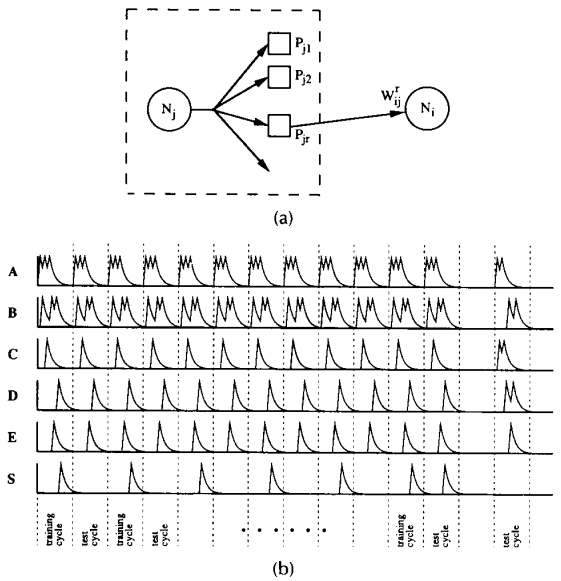


Fig. 3. An expanded neuron model for complex sequence recognition. b) Recognition of the complex sequence $\mathcal{S}_3$: *A-B-A-C-A-B-E-B-D*. Ten neurons have been modeled, and temporal activities of $N_1, N_2, \cdots, N_5$, and N_{10} are displayed in the figure from top to bottom. During each cycle of training a peak of activity indicates the activation of the corresponding neuron. Thus, for example, the trace for *A* has 3 peaks, and the trace for *C* has 1 peak between the second and third peaks of *A*. Each training cycle is followed by a test cycle during which the sequence detecting neuron is not activated externally. After 6 trainings, the sequence detecting neuron *S* can be activated by the presentation of $\mathcal{S}_3$ alone. The last column is to test if the detecting neuron can be activated by another sequence, *A-C-A-C-D-B-E-D-B*. The parameters are: $c = 5$, $\alpha = 0.4$, $C_i = 2.0$ $(i = 1, \cdots, 10)$, $\Gamma_{10} = 0.629$, $\Delta = 1$.

most recent impulse generated by N_j. Thus, the way the P_{jr} operate is like a stack: A new impulse generated by N_j pushes the whole array P_{jr} by one place and imprints itself on the first terminal (see the definition below). Every P_{jr} is decremented in each cycle Δ. In the following model, the membrane potential and the output of neuron N_i at time t are $m_i(t)$ and $S_i(t)$, respectively.

$$P_{jr}(t) = \begin{cases} (1-\alpha)P_{jr}(t-\Delta) & \text{if } S_j(t) = 0 \\ 1 & \text{if } S_j(t) = 1 \text{ and } r = 1 \\ (1-\alpha)P_{j,r-1}(t-\Delta) & \text{if } S_j(t) = 1 \text{ and } r > 1 \end{cases} \tag{11}$$

$$m_i(t+\Delta) = \sum_{j \neq ir=1}^{c} \sum W_{ij}^r p_{jr}(t) + I_i(t) \tag{12}$$

$$S_i(t) = f[m_i(t) - \Gamma_i] \tag{13}$$

$$\begin{cases} \hat{W}_{ij}^r(t) = W_{ij}^r(t-\Delta) + C_i S_i(t) P_{jr}(t) \\ W_{ij}^r(t) = \hat{W}_{ij}^r(t) \Big/ \sum_{j' \neq ir=1}^{c} \sum \hat{W}_{ij'}^r(t) \end{cases} \tag{14}$$

where α is the decay parameter of $p_{jr}(t)$, playing a similar role as the $g(x)$ introduced earlier, and c represents the number of terminals of each neuron N_j. W_{ij}^r is the weight of the synapse that the rth terminal of N_j makes on N_i. It is sufficient to set c to the number of maximal occurrences of a symbol in a sequence, for example, $c = 3$ is sufficient to recognize $\mathcal{S}_3$. The choice of c—the number of terminals each neuron has—limits the number of occurrences of the same symbol within a temporal sequence. Synaptic modification is defined in the same way as before, except that we have c synapses between two neurons instead of one. Due to the normalization in (14), weights are set initially $W_{ij}^r(t) = 1/[c(n-1)]$.

The maximization principle applies to this model for complex sequence recognition similarly. The conclusions from the previous section (Theorems 1, 2, and 3, Corollaries 1 and 2) are also established in this model. Particularly, we have

Theorem 4. After neuron N_i has learned any complex sequence $\mathcal{S}_i$, the presentation of $\mathcal{S}_i$ induces the maximal postsynaptic potential in N_i.

If we set Γ_i in the same way as in the previous section, this theorem guarantees that system (11)–(14) can learn and recognize any complex temporal sequences without mistake. Figure 3 shows a simulation for learning and recognizing sequence $\mathcal{S}_3$.

This neural model can be used directly for the recognition of temporal sequences which contain distortions. This can be achieved in the following two steps:

1) Lowering the threshold of each sequence-detecting neuron. The previous threshold setting (compare Eq. (9)) is only appropriate in the absence of distortions. In order to tolerate distortions, we need to lower the threshold a little bit such that if the current sequence induces a membrane potential close to that induced by the learned sequence, the corresponding neuron will fire. Thus a sequence-detecting neuron can be fired by a set of sequences close to the learned sequence.

2) In case 1, a currently presented sequence may activate more than one detecting neuron, which is not desirable. To avoid this situation, we can feed the signals of all the firing neurons (if any) to a competitive neural network (winner-take-all network, see [18], [19]). This competitive network ensures that only the neuron which is activated and has the biggest signal is activated.

With this extension, the model can serve as a general sequence recognizer. Figure 4 shows the whole system architecture. In the input layer each neuron represents a spatial pattern. The connection from the input layer to the recognizer layer is an all-to-one correspondence. For clarity only one neuron is shown in the recognizer layer. The connection from the recognizer layer to the competition layer is a one-to-one correspondence.

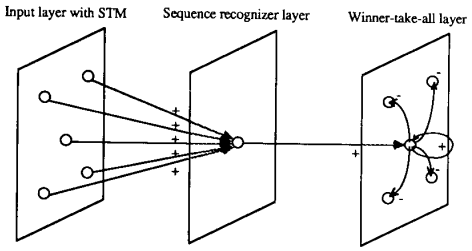


Fig. 4. General architecture for temporal sequence recognition.

III. Temporal Sequence Reproduction

Temporal sequence reproduction is a different, and somehow more difficult, issue than temporal sequence recognition. If the sequence in question is a simple sequence, reproduction becomes much easier because we only need to store transitions between each two consecutive patterns. This, as mentioned in the introduction, has been achieved by many authors. However, the real difficulty lies in reproduction of complex sequences, where a correct transition to a pattern is determined by its *context*, not simply by one previous pattern.

The model for sequence recognition can be borrowed for sequence reproduction. In the previous models, each neuron in the input layer represents a single spatial pattern. But a neuron can also be considered as a sequence detector. It appears that if we make a neuron function as a detector of the part of a sequence before the pattern that this neuron represents, we can readily realize sequence reproduction. This idea has the attractive feature that sequence training for reproduction is nothing but a simple sequence presentation, not like sequence recognition where we need to activate (teach) the sequence-detecting neuron deliberately for each sequence detector. However, it has some problems for general complex sequence reproduction. First, a *self-reference problem* occurs when transition to a pattern depends on a context which contains the pattern itself, like pattern *A* in the four-degree sequence: *A-B-A-C-A-B-A-D*. Second, since we usually choose a memory span equal to or a little larger than the degree of the sequence (we cannot expect a memory span of the same length as the sequence itself), this requires that a neuron be able to be activated by different subsequences. For example, in the 2-degree sequence *A-B-C-A-D-E-A-F-G-A-H-I*, pattern *A* can be transitioned from *B-C*, *D-E*, and *F-G*. This violates the previous hypothesis that one neuron can only be activated by one sequence. We call this the *multiple reference problem*. These two problems can be solved with a direct extension if sequences to be reproduced are not more complex than first-order complex sequences.

The idea proposed here separates the neurons for detection from those standing for spatial patterns. An additional layer of neurons, layer *d*, is used as sequence detectors which follow the model for complex sequence recognition presented previously. This layer is connected to the original layer of spatial patterns (layer *p*) bidirectionally such that a sequence detector can activate the appropriate next spatial pattern. This connection configuration is shown in Figure 5. The number of neurons in layer *d* must not be less than the length of the sequence minus the degree of the sequence. Let η be the degree of the sequence. During

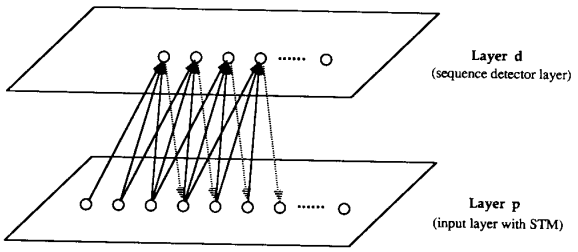


Fig. 5. Network architecture for temporal sequence reproduction. At the beginning, the connections between layer *p* and layer *d* are all-to-all correspondence. The appropriate connection pattern for reproduction will emerge after repetitive training with a temporal sequence.

training a sequence is presented to layer *p*. After the first $\eta - 1$ patterns have been presented in layer *p*, a neuron in layer *d* is randomly chosen (but fixed in successive trainings) to fire synchronously with each presentation of a spatial pattern in layer *p*. The recurrent connections from layer *d* to *p* are set up according to the Hebbian rule. That is, whenever there is a neuron N_i^d firing in *d* and N_j^p firing in *p*, there will be a synaptic link established from N_j^d to N_i^p. This training process is repeated several times until each neuron in layer *d* has learned to recognize a specific subsequence. Later reproduction of the sequence is done by presenting its initial context to layer *p*. Since our sequence detection model can detect any complex sequence, this model can reproduce any complex sequence.

As an example, we have trained this three-layer network with the four-degree higher-order complex sequence $\mathbb{S}_4$: A-B-A-C-D-A-B-A-E-F-A-B-A-G-H-A-B-A-I-J. During training, sequence $\mathbb{S}_4$ is presented to layer *p*. After the first three patterns have been presented in layer *p*, a neuron in layer *d* is chosen to fire synchronously with each presentation of a spatial pattern in layer *p*. After six trainings, the network can reproduce the whole sequence by being presented *A-B-A-C*, the initial context. Figure 6 shows this process,

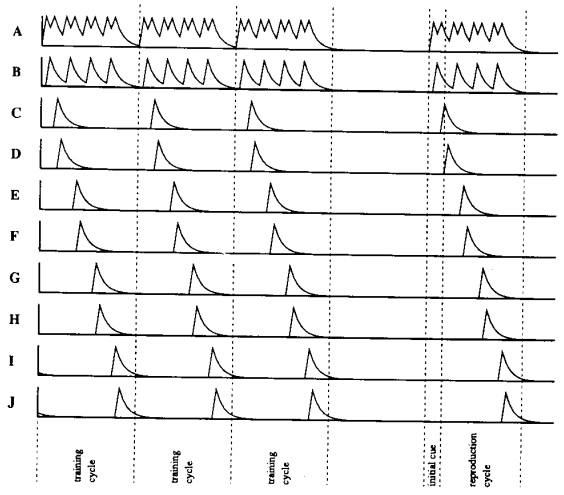


Fig. 6. Reproduction of a 4-degree higher-order complex sequence $\mathbb{S}_4$: *A-B-A-C-D-A-B-A-E-F-A-B-A-G-H-A-B-A-I-J*. The model is trained with this sequence for 6 times and the last 3 trainings are shown in the figure. The entire sequence is reproduced by the model with the presentation of $\mathbb{S}_4$'s initial context: *A-B-A-C*.

which includes the last three trainings with the reproduction.

IV. Conclusion

Two ideas are central to this neural model of temporal sequence learning: short-term memory and sequence sensitivity.

In order to link two temporally discontiguous patterns, the earlier pattern has to be preserved for a certain period of time, which is assumed to be achieved by STM in our model. As discussed, STM is modeled here by recurrent excitatory connections. Our results suggest an important computational function of STM, that is, STM could lay a basis for temporal sequence learning. In addition, STM span is correlated to the degree of a sequence in the model. One of the predictions from this model is that the STM capacity puts a direct restriction on the degree of a primitive sequence to be learned. Hierarchical methods may be required for learning sequences with greater degrees than STM capacity, as suggested in the chunking theory [20]. A recurring subsequence may be viewed as a single unit, which could lessen STM load significantly.

The second key idea is sequence-sensitive training. We find that during training that follows the Hebbian learning rule and the normalization of all synaptic weights, the input potential to a neuron increases monotonically (compare Corollary 2). This idea underlies the recognition as well as the reproduction of complex temporal sequences. In the model, the transition from a previous subsequence to a current pattern is not stored explicitly anywhere in the network, as in many other modeling efforts. Instead it is reflected by the distribution of weights to a sequence-detector neuron, and this distribution will maximize the input potential of the detector neuron upon the presentation of the previous subsequence. One important feature of this model is that the length of the previous subsequence (or the degree of a complex sequence in general) does not affect the performance of the sequence learning, whereas it could cause severe problems for many other models previously proposed. The training turns a neuron from sequence insensitive to sequence sensitive, like order emerging from chaos. A graded input to a synapse can be viewed as a firing rate of impulses or a graded potential, and the Hebbian rule and the normalization rule are both biologically plausible. So this sequence-sensitive training is consistent with the known neurobiology.

The model aims at dealing with complex temporal sequences directly, not only because they are indispensable for real applications such as speech recognition, music generation, and so on, but also they pose critical problems for previous models for temporal sequence learning. This model provides a general solution to this problem both for sequence recognition and sequence reproduction without causing significant extra computational expense. At the same time, the problem of *ad hoc* wiring for temporal coupling, existing for example in the outstar avalanche model and the selection model, has been avoided here.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA*, vol. 79, pp. 2554-2558, 1982.

[2] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, "A learning algorithm for Boltzmann machines," *Cognitive Science*, vol. 9, pp. 147-169, 1985.

[3] S. Grossberg, "Some networks that can learn, remember, and reproduce any number of complicated space-time patterns, I" *J. Math. and Mechan.*, vol. 19, pp. 53-91, 1969.

[4] C. Stanley and W. L. Kilmer, "A wave model of temporal sequence learning," *Int. J. Man-Machine Stud.*, vol. 7, pp. 397-412, 1975.

[5] B. Kosko, "Bidirectional associative memory," *IEEE Trans. Sys. Man. Cybern.*, vol. 18, pp. 49-60, 1988.

[6] T. Dehaene, J. P. Changeux, and J. P. Nadal, "Neural networks that learn temporal sequences by selection," *Proc. Natl. Acad. Sci. USA*, vol. 84, pp. 2727-2731, 1987.

[7] D. Kleinfeld, "Sequential state generation by model neural networks," *Proc. Natl. Acad. Sci. USA*, vol. 83, pp. 9469-9473, 1986.

[8] H. Sompolinsky and I. Kanter, "Temporal association in asymmetric neural networks," *Phys. Rev. Lett*, vol. 57, pp. 2861-2864, 1986.

[9] D. W. Tank and J. J. Hopfield, "Neural computation by concentrating information in time," *Proc. Natl. Acad. Sci. USA*, vol. 84, pp. 1896-1900, 1987.

[10] J. Buhmann and K. Schulten, "Noise-driven temporal association in neural networks," *Europhysics Letters*, vol. 4, pp. 1205-1209, 1987.

[11] H. Gutfreund and M. Mezard, "Processing of temporal sequences in neural networks," *Phys. Rev. Lett.*, vol. 61, pp. 235-238, 1988.

[12] 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.

[13] D. O. Hebb, *The Organization of Behavior.* Wiley: New York, 1949.

[14] S. Grossberg, "Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors," *Biol. Cybern.*, vol. 23, pp. 121-134, 1976.

[15] R. Lara, M. A. Arbib, and A. S. Cromarty, "The role of the tectal column in facilitation of amphibian prey-catching behavior: a neural model," *J. Neurosci.*, vol. 2, pp. 521-530, 1982.

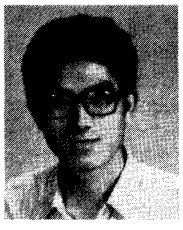
[16] D. L. Wang, J. Buhmann, and C. von der Malsburg, "Pattern segmentation in associative memory," *Neural Computation*, vol. 2, pp. 94-106, 1990.

[17] C. von der Malsburg, "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, pp. 85-100, 1973.

[18] S. Amari and M. A. Arbib, "Competition and cooperation in neural nets," in *Systems Neuroscience*, J. Metzler, Ed. New York: Academic Press, 1977, pp. 119-165.

[19] D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," in *Parallel Distributed Processing*, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: The M.I.T. Press, 1986, vol. 1, pp. 151-193.

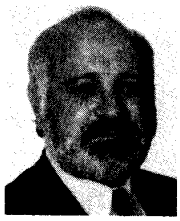
[20] H. A Simon, *Models of Thought.* New Haven, CT: Yale University Press, 1979.



DeLiang Wang was born in Anhui, the People's Republic of China on January 27, 1963. He received the B.S. degree and the M.S. degree in computer science from Beijing University, Beijing, China, in 1983 and 1986, respectively.

From July 1986 to December 1987 he was with Institute of Computing Technology, Academia Sinica, Beijing, China. He is currently completing a Ph.D. degree in computer science at University of Southern California, Los Angeles, CA.

His present research interests include neural mechanisms of visuomotor coordinations temporal sequence learning using neural networks, and visual pattern perception.



Michael A. Arbib was born in England in 1940 and grew up in Australia. He received the B.Sc. (Hons.) degree from Sydney University, and received the Ph.D. in mathematics from MIT in 1963.

After five years at Stanford, he became chairman of the Department of Computer and Information Science at the University of Massachusetts at Amherst in 1970, and remained in the Department until August, 1986, helping found the Center for Systems Neuroscience, the Cognitive Science Program, and the Laboratory for Perceptual Robotics, for each of which he served as Director. He is currently a Professor of Computer Science, Neurobiology, Physiology, Biomedical Engineering, Electrical Engineering, and Psychology at the University of Southern California, where he is the founder and first Director of the Center for Neural Engineering.

Dr. Arbib's most recent books are *The Metaphorical Brain 2: Neural Networks and Beyond* (Wiley-Interscience, 1989), *Dynamic Interactions in Neural Networks: Models and Data* (M. A. Arbib and S. Amari, Eds., Springer-Verlag, 1989), and *Visuomotor Coordination: Amphibians, Comparisons, Models and Robots* (J.-P. Ewert and M. A. Arbib, Eds., Plenum Press, 1989).