Timing and Chunking in Processing Temporal Order

DeLiang Wang and Michael A. Arbib

Abstract—**A computational framework of learning, recognition and reproduction of temporal sequences are provided, based on an interference theory of forgetting in short-term memory (STM), modeled as a network of neural units with mutual inhibition. The STM model provides information for recognition and reproduction of arbitrary temporal sequences. Sequences are acquired by a new learning rule, the attentional learning rule, which combines Hebbian learning and a normalization rule with sequential system activation. Acquired sequences can be recognized without being affected by speed of presentation or certain distortions in symbol form. Different layers of the STM model can be naturally constructed in a feedforward manner to recognize hierarchical sequences, significantly expanding the model's capability in a way similar to human information chunking. A model of sequence reproduction is presented that consists of two reciprocally connected networks, one of which behaves as a sequence recognizer. Reproduction of complex sequences can maintain interval lengths of sequence components, and vary the overall speed. A mechanism of degree self-organization based on a global inhibitor is proposed for the model to learn required context lengths in order to disambiguate associations in complex sequence reproduction. Certain implications of the model are discussed at the end of the paper.**

I. Introduction

TEMPORAL ARRANGEMENT is at the heart of thought, language and action, and contributes greatly to human intelligence. Recognizing temporal patterns is crucial in hearing and vision, and generating the temporal patterns underlies processes like motor pattern generation, speech and singing. The basic function of sequence generation is to reproduce learned sequences. This article presents a computational theory of temporal order, based on interaction and integration of local neuron populations (the basic functional units). The proposed neural circuitry recognizes and reproduces any complex temporal sequence, and can compensate for a range of distortions in time (time-warp problem) and in form (erroneous symbols).

Following the terminology introduced by Wang and Arbib [63], a temporal sequence S is defined as

$$p_1 - p_2 \cdots - p_N$$

and each p_i is called a component of S (sometimes a spatial pattern, or just a symbol). The length of a sequence is the number of components in the sequence. Any $p_i - p_{i+1} - \cdots - p_j$, where $1 \le i \le j \le N$, is called a subsequence of S. If S contains repetitions of the same subsequence, like $A - B$ in $C - A - B - D - A - B - E$, it is called a complex sequence, otherwise a simple sequence. The context of the current symbol p_i in a complex sequence S is the prior subsequence required to cue p_i unambiguously. The degree of p_i is the length of this context. The degree of a sequence is the maximum degree of its components. Therefore, a simple sequence is equivalent to a 1-degree sequence.

Early models of neural networks to store, recognize and reproduce a temporal sequence of input stimuli include the outstar avalanche [20] and the wave model [57] that can reproduce a sequence of patterns based on learned associations between consecutive patterns. More recently, using a synaptic triad made up of three neurons as building blocks (high-order synapses), Dehaene *et al.* [11] proposed a layered neural network, called the selection model, which can recognize temporal sequences. Kosko's [36] bidirectional associative memory built from two neural fields can reproduce a sequence of patterns that alternates between the two fields.

Storage of temporal sequences in the spin-like Hopfield network has been proposed by several authors [30], [55], [59], [5], [22], [23], [37], [25]. 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. Storage and retrieval of complex sequences is difficult since, in most of these models, a given pattern can occur only once among all the stored sequences. Recognition and reproduction of temporal sequences have also been explored using the backpropagation network ([28], [14], [18], [62], [39], [2], [48], among others). There are two basic architectures behind most of the models: In the Jordan network [28] the output layer associated with a pattern is fed back and blended with the input representing the next pattern, whereas in the Elman network [18] the hidden layer is fed back to influence the next pattern. Complex sequence recognition and reproduction again cause severe problems to this type of model. Some remedies have been proposed for dealing with complex sequences for such models, and will be discussed later in this paper. One popular scheme to recognition has been to construct a buffer to hold a fixed number of the most recent elements of the input sequence. Implemented by fixed delay lines, the buffer turns a temporal recognition problem into a spatial recognition problem, and backpropagation is employed for training [62], [39]. Although some success has been achieved in small sets of presegmented temporal patterns, the approach has serious drawbacks in terms of efficiency, which we will come back to in the discussion.

Manuscript received February 7, 1992; revised October 3, 1992. This work was supported in part by grant 1RO1 NS 24926 from the National Institutes of Health (M.A.A., Principal Investigator), and in part by an NSF Research Initiation Award (IRI-9211419) (to the first author).

D. Wang is with Department of Computer and Information Science and Center for Cognitive Science, The Ohio State University, Columbus, OH 43210-1277, USA.

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

IEEE Log Number 9207515.

In a recent paper [63], we proposed a new mechanism for learning temporal sequences. We modeled short-term memory (STM) by units comprising recurrent excitatory connections between two local neuron populations. Each population is represented by a single quantity corresponding to local field potential. The activity induced by an input signal to a unit oscillates with damping, thus decaying over time. Using a Hebbian learning rule at each synapse and a normalization rule among all synapses to a unit, the neural networks with this model of STM are able to learn complex temporal sequences, recognize these sequences with tolerance to certain distortions in form, and reproduce them. What distinguishes our model from others are two basic hypotheses embodied in the model: 1) There is a common mechanism to process both complex sequences and simple sequences; and 2) Reproduction of a component in a sequence is based on recognition of the context of the component.

STM was modeled by decay with a fixed temporal course that makes the previous model unable to handle the time-warp problem. For sequence recognition, we wish a network to recognize a time-warped sequence, whereas for reproduction we wish a network to reproduce a sequence with the same temporal course as the training sequence. In addition, we wish that recognition is not affected by changes in rates of presentation, and at the same time, reproduction can vary its overall speed. We attempt to solve the time-warp problem in the present study. In the previous model, all context detectors assume the same degree, which must not be less than the degree of the entire sequence to be recalled. The requirement is replaced in this paper by a dynamic tuning mechanism whereby each detector learns during training its necessary degree for unambiguously producing the next symbol. We also propose a mechanism for hierarchical sequence recognition, similar to human information chunking. This mechanism seems both natural and necessary for processing long sequences, like a paragraph of sentences, a piece of music, and so forth.

II. A Computational Model of STM

It has been found in the study of memorizing nonsense syllables that each syllable in the series has links not only to adjacent words in the series, but also to remote words [40]. In order to link two temporally discontiguous patterns, the previous one has to be memorized until the latter one occurs. This typical short-term memory phenomenon lays an important basis for temporal order processing. In order to provide for temporal processing, a model of STM must provide the following four basic functions:

1) Maintaining a symbol for a short time period. How long can an item be retained? Peterson and Peterson [47] found that the probability of a correct recall declined rapidly over an 18-second period, when subjects were asked to perform some distracting task to prevent rehearsal. What causes forgetting? Two dominant views are decay versus interference [49]. An interference theory proposes that memory for other material or the performance of another task interferes with memory and thus causes forgetting. A decay theory, on the other hand, proposes that forgetting still occurs even if the subject had to do nothing over the retention interval, as long as the subject did not rehearse the material.
2) Maintaining a number of symbols. Miller [44] tells us that the capacity of STM is only about seven symbols, but suggests that recoding information to form chunks can help overcome this limitation.
3) Coding the order of input symbols. Given that STM can hold several items simultaneously, the order that these items enter STM must also be coded some way. It has been observed that subjects engage in linear scanning when judging whether a test symbol is contained in a short memorized sequence [58]. However, it remains unknown how order is coded in STM.
4) Coding the length of presentation of each symbol. When one learns a sequence, one can recognize it even though each component of the sequence is presented at a different speed. Yet, a professional musician can recall a multiple-page score, reproducing almost exactly the memorized length of each note, although each note may last differently. Since STM is an interface between input symbols and long-term memory (LTM), STM must be able to code the length of each held symbol. This function of STM provides first level information for solving the time-warp problem.

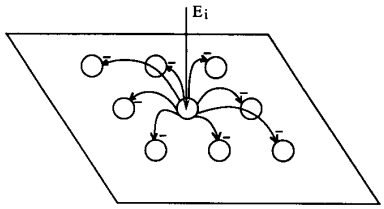


Fig. 1. Diagram of the STM model. Unit i receives external input E_i as well as inhibition from all other units in the model. The figure shows only outgoing projections from unit i. Minus signs indicate inhibition.

Our previous STM model conforms with the decay theory of forgetting, since the activity of a unit when stimulated oscillates and decays over time. The order of input symbols is coded by the different amplitudes of unit activities elicited by these symbols because of decay over different times since presentation. However, the number of items the STM model can hold varies with the length of the presentation intervals of each symbol, and the longer each presentation takes the fewer items can be held in STM. Furthermore, the model cannot code the length of each symbol presentation, and therefore it fails to solve the time-warp problem.

Waugh and Norman [65] report findings that clearly favor the interference theory. The current majority view seems to weight interference more heavily than decay. Although some decay may occur [10], the amount of forgetting caused by decay is substantially less than the amount caused by interference [49], [45]. The following computational model of STM we will describe is based on the interference theory.

Let unit i represent the ith local neuronal population ($i = 1, 2, \cdots, n$), the building block of this STM model, and x_i

its excitation level. Each unit receives an external input E_i, which is 1 so long as the external input is on and 0 otherwise, and is inhibited by all the other units, as shown in Fig. 1. Two further quantities are associated with unit i: the internal state, s_i, which signals activation of the unit and provides inhibition to the other units; and the excitation level x_i, which provides a decaying memory trace, and which is used in the learning rules of the next section. The internal state s_i is defined as

$$s_i(t) = \begin{cases} 1, & \text{if } E_i(t) = 1, E_i(t-1) = 0 \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

From the definition we can see that the internal state is activated only by the beginning of an external input. Detection of the beginning part of an external input can be neurally implemented with a threshold and adaptation of the external input.

The excitation level of unit i lies in the range of $\{0, 1, \cdots, T\}$, and is defined as

$$x_i(t) = \begin{cases} T & \text{if } s_i(t) = 1, \\ x_i(t-1) - 1 & \text{if } x_i(t-1) > 0, y_i(t) = 1 \\ x_i(t-1), & \text{otherwise,} \end{cases} \quad (2)$$

where y_i represents overall inhibition that unit i receives from the other units and is formulated as[1]

$$y_i(t) = f(\sum_{j \neq i} s_i(t-1) - 1) \quad (3)$$

with

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

so that $y_i(t) = 1$ iff an external input is applied to any unit other than unit i. From the above definitions we see that whenever $s_i(t) = 1, x_i(t)$ is brought to its highest value T and unit i is activated or triggered. If any of the units is activated, the inhibition that it exerts on the rest of the network will drive all other active units, i.e., those whose excitation levels are larger than 0, down to the next lower level.

Now let us see how this model of STM satisfies the above four requirements. First, this model preserves a symbol, or an information item, on a unit whose excitation level codes the item. Let us assume that external inputs arrive at STM serially (it is easy to serialize simultaneous inputs by a competitive network, see among others [12], [21], [1], [51]. Any new item input to STM decrements the excitation levels of all active units in STM. Therefore STM can at most code T items. T is a system constant that is equivalent to the capacity of the STM model, suggesting that T be about 7 ± 2 in a model of humans [44]. Variability in capacity may attribute to individual differences and different types of materials to be memorized. If we consider only the case where all items in STM are different for the time being, then a symbol can be maintained in STM from when it is input to when the Tth subsequent item is entered, conforming with the interference theory. Secondly, T symbols can be preserved simultaneously in the model. Thirdly, the order of input symbols is coded by the excitation levels of the units that represent the symbols. The larger the excitation level of a unit is, the more recent is the symbol represented by the unit. Since all inputs to the STM model are serial, there is a strict temporal order among all symbols held in the model. Finally, the length of a symbol's presentation is reflected by the time period while the corresponding external input is on, and its coding mechanism will be given later. In conclusion, the above simple formal model is capable of coding the four necessary functions of STM. Possible neural circuitries for implementing units, local neuronal populations, will be discussed in the discussion section. We will see in the following sections how information carried in the model is used for processing temporal order.

[1] Since the weights of inhibitory connections are the same, the mutual inhibitory connections can be replaced by a global inhibitor. A global inhibitor can reduce the number of connections by one order of magnitude, but results in a less reliable system due to information centralization on the inhibitor.

III. General Sequence Recognition

The following model for general sequence recognition is based on the above STM model, and the learning algorithm is basically the same as used previously [63]. The major focus, compared to previous work, is the time-warp problem. Here, sequence recognition is not affected by varying presentation intervals for individual components in a sequence, which property is called *interval invariance.*

A. Simple Sequence Recognition

Before we propose a solution for general sequence learning, it helps elucidate basic ideas by presenting a model for simple sequence recognition. Suppose that an extra unit 0, called a detector, is to be trained for recognizing a simple sequence S_0. The detector unit receives projection from n units in the STM model, and s_0 is formed by

$$s_0(t) = f(\sum_{i=1}^{n} W_{0i} x_i(t-1) + I_0(t-1)\Gamma_0) \quad (5)$$

where W_{0i} is the connection weight from unit i to the detector; Γ_0 is the threshold of the detector, and I_0 represents the external input to the unit. Learning, or modification of connection weights, follows a Hebbian rule [24] with normalization [43]

$$\begin{cases} \hat{W}_{0i}(t) = W_{0i}(t-1) + C_i s_0(t) x_i(t) \\ W_{0i}(t) = \hat{W}_{0i}(t) / \sum_{i'=1}^{n} \hat{W}_{0i}^{i'}(t) \end{cases} \quad (6)$$

where C_i is a gain factor of learning. The larger is C_i, the faster is learning and the more easily is the memory value overwritten by a new stimulus. The effect of learning on the detector is to change the distribution of all weights to that unit, so it is reasonable to assume that initially $W_{0i} = 1/n$.

Let $S_0 = p_{0_1} - p_{0_2} - \cdots - p_{0_i} - \cdots - p_{0_K}, 1 \leq 0_i \leq n$. Without loss of generality, we suppose that pattern p_{0_i} fires (is represented by) unit 0_i. Since S_0 is a simple sequence, $0_i \neq 0_j$ if $i \neq j$. Our purpose is to train the detector to recognize S_0 with interval invariance. Training is done by presenting S_0 to the model and activating unit 0, i.e., setting $I_0(t)$ above the threshold Γ_0, immediately after the presentation of S_0. The end of a sequence presentation is detected by an end detector in the system, which uses indications like pauses (implicit) or separators (explicit) between sequences.

We call this specific type of training *attentional learning*. It is different from unsupervised learning, and is also different from typical supervised learning where a desired output or an answer (as in reinforcement learning) has to be provided externally. The activation of a sequence detector at the end of presentation of the sequence may be driven by attention, which is indispensable for learning a sequence [46], [8]. In this learning paradigm, we say unit 0 is *attended* when I_0 is brought by the system above Γ_0.

During training of S_0, each presentation is allowed to vary its speed. That is, any p_{O_i} can have a different presentation interval from that of any other component of S_0 of the same presentation trial, and even from that of the same p_{O_i} of a different trial. If the detector can be activated by presentation of S_0 but not by any other sequence, we say that it has learned to recognize the sequence. Of course, for recognition to be interval invariant, after learning the detector should also be activated by the same S_0 with a presentation speed different from any used in training.

The input potential IP_0 of S_0 to the detector is defined to be the weighted sum to the unit at time t' immediately after the presentation of S_0, that is

$$IP_0 = \sum_{i=1}^{n} W_{00_i} x_{0_i}(t') = \sum_{i=1}^{K} (T - K + i) W_{00_i}. \quad (7)$$

According to (2), $x_{0_i}(t)$ is set to T by input p_{0_i}, and decrements only when a new input is received. Thus, $x_{0_i}(t')$ equals $T - K + i$. Equation (7) is the same as (7) in Wang and Arbib [63], except that the function $g(l)$ there is instantiated to a linearly decreasing function of (2) here. The formal analysis in that paper applies as long as g is monotonically decreasing and so all relevant theorems and corollaries are also established in this model, and are summarized below without proof.

1° Repeated training with S_0 leads all weights to unit 0 to have the distribution: $W_{00_i} = 2(T-K+i)/[K(2T-K+1)]$, with $W_{0j} = 0$ for $j \neq 0_1, \cdots, 0_K$

2° Repeated training with S_0 leads to

$$IP_0 = \frac{2}{K(2T - K + 1)} \sum_{i=1}^{K} (T - K + i)^2 \quad (8)$$

where IP_0 depends only on the length, K, of the sequence.

3° Define ΔIP_0^m as the IP_0 after the mth presentation of S_0 minus the IP_0 after the $(m-1)$st presentation of S_0. Then

$$\Delta IP_0^m = \frac{\Delta IP_0^1}{Q^{m-1}} \quad (9)$$

where $Q = 1 + C_i(2T - K + 1)/2$. Furthermore, if repeated training with S_0 begins with the initial condition, i.e., $W_{0i} = 1/n$, then after the first training, $\Delta IP_o^1 > 0$. Therefore, based on (9), $\Delta IP_0 > 0$ after each training. In other words, IP_0 increases monotonically with sequence training.

The above conclusions imply that if we set Γ_0 in (5) to the input potential expressed in (8), i.e.,

$$\Gamma_0 = \frac{2}{K(2T - K + 1)} \sum_{i=1}^{K} (T - K + i)^2 \quad (10)$$

then the result of training is to build up IP_0 so as to fire the detector by presentation of S_0. Since Γ_0 in (10) is the limit value of IP_0, a small error ε should be subtracted from that Γ_0 when applied in practice. Because Γ_0 is dependent only on the length K of the sequence in question, it can be easily set up during the first training of the sequence.

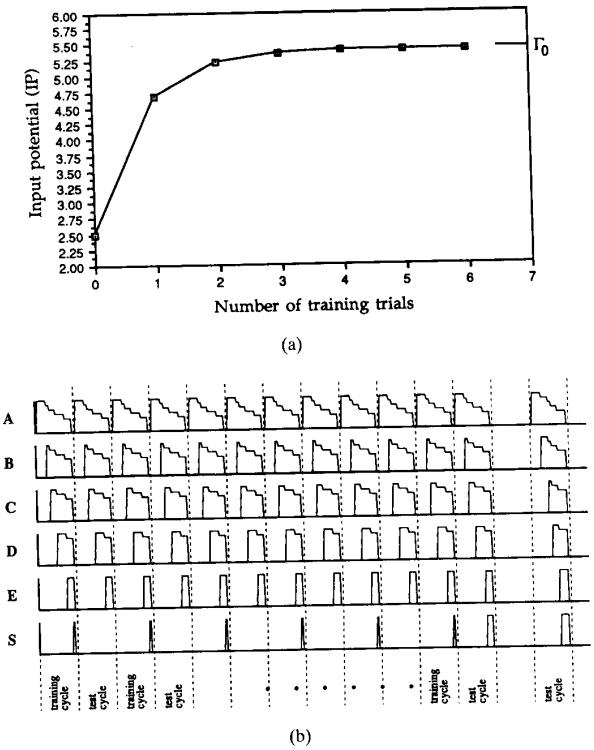
4° After the detector has learned sequence S_0, only presentation of S_0 induces the maximum activity on the detector unit.

The result embodies a maximization principle that repeated training of a sequence polarizes the weights of the corresponding detection unit so that it can only be activated by this specific sequence. The maximization principle has two parts. The first involves training with (6) that drags the weight distribution of the detector along with the direction of the training signals from units activated by the sequence. The second simply uses the fact that the inner product of two normalized parallel vectors reaches the maximum value. The latter fact has been previously used for pattern classification (the nearest neighbor method, see [16]) and even in pattern recognition by neural networks [38]. Our contribution lies in proposing a biologically plausible learning scheme ((6)) that naturally prepares weights for later application of the maximization process.

A computer simulation of this simple sequence recognition with interval invariance was conducted, and the result is shown in Fig. 2. The sequence to be detected was $A-B-C-D-E$, each component being presented for different intervals that are created by a random number generator within a certain range. Fig. 2(a) shows the monotonic increase of IP_0 with number of sequence presentation trials. The increase follows a typical inverse exponential curve with increase rate exponentially decreasing. Fig. 2(b) depicts the actual training and recognition process, with training intervals $\{9, 3, 6, 9, 5\}$ for A, B, C, D, E respectively. After the sixth training trial, IP_0 went above the system-set threshold, and so any following presentation of the sequence was able to activate the detector. After that, the same sequence with the different interval series $\{9, 7, 3, 6, 4\}$ (also generated by the random number generator) for its components was tested, and the model succeeded in recognizing this time-warped sequence. See the figure legend for the parameter values used.

The gain parameter C_i in (6) controls the overall speed of learning, and it is imaginable that with a very large C_i the system exhibits so called one-shot learning: imprinting a sequence on a detector after the first presentation. One-shot learning has been previously demonstrated in neural systems such as the ART pattern recognizer [6]. It seems more efficient to have one-shot learning than gradual learning, since one detector is dedicated to one sequence. We provide a general learning rule of (6) to account for both gradual and one-shot learning. One advantage of gradual learning is its reliability in the sense that a sequence detector is less prone to damage due to wrong "attention" (system activation) paid to the detector.

The idea behind interval invariance is that during presentation of a sequence component, only the beginning of presentation of its input symbol triggers activity on a unit, and only presentation of a new symbol to the network decrements



(a)

(b)

Fig. 2. (a) Monotonic increase of input potential IP0 with number of training trials of sequence A-B-C-D-E. After the sixth trial, IP_0 was within a small error $\epsilon = 0.001$ of the system-set threshold value of unit 0 ($\Gamma_0 = 5.4$). (b) Training for recognition of the sequence with time-warping. Let units 1-5 represent patterns A, B, C, D, E respectively; a symbol in the figure indicates the corresponding unit. S corresponds to the detector unit 0. During each training cycle, the sequence was presented, followed by an activation of the detector unit (the attentional learning rule). The activation is indicated in the figure by a peak value equal to T. Each training trial was followed by a test cycle, during which the sequence was presented alone in order to see if unit 0 could be activated by the sequence. Presentation intervals for individual components were generated by a ranged random number generator, and they were {9, 3, 6, 9, 5} for A, B, C, D, E respectively. After six trials, the detector unit was able to be activated by another presentation of the sequence. After the detector learned the sequence, another test trial was made by presenting the same sequence with a different interval series {9, 7, 3, 6, 4} also randomly generated. As shown in the last column, the detector recognized the time-warped sequence. The parameters are: $n = 10$, $C_i = 0.04$ ($i = 1, \cdots, 10$), $T = 7$.

its current activity levels—i.e., the only decay is triggered by interference. Therefore it does not matter how long that presentation lasts. This same idea is used for recognition of any complex sequence, as presented next.

B. Complex Sequence Recognition

The above mechanism for simple sequence recognition cannot be directly applied for complex sequence recognition. A unit corresponds to a symbol in a sequence, and the external activity of the unit is represented by only one quantity: its excitation level. Therefore according to (2) a later occurrence of a symbol in a sequence may overwrite an earlier occurrence stored in the STM model. For example, the different occurrence of A in sequence $S_1 : A-B-A-C-A-B-E-B-D$. To overcome this problem, we proposed [63] that a unit was represented by an expanded network, such that it has multiple terminals to hold different occurrences of a symbol, with multiple channels connecting to other units. Fig. 3(a) shows a diagram for a single unit. The following model combines this idea for solving the overwriting problem with the new STM model for interval invariance.

Suppose unit i has m terminals, and the excitation level of its rth terminal is represented by x_{ir}. A new input maximally activates x_{i1}, and "shifts" all other traces "downwards," so that x_{ir} holds the rth most recent occurrence of the symbol represented by the unit. The STM model ((1)–(4)) and the definitions of E_i, s_i, and y_i thus remain the same except that which is shown in (11) at the bottom of the page.

Again let unit 0 be trained to detect an arbitrary sequence S_0. As before, during training each component of the sequence is allowed to have a different presentation interval from that of any other component of S_0 of the same trial, and from that of the same component on a different trial. After learning, the model should be able to recognize S_0 presented with a speed different from that of any training trial. The detector receives inputs from n units from the STM model, and s_0 is defined as

$$s_0(t) = f\Big(\sum_{i=1}^{n}\sum_{r=1}^{m} W_{0i}^r x_{ir}(t-1) + I_0(t-1) - \Gamma_0\Big) \quad (12)$$

where W_{0i}^r is the weight of the connection that the rth terminal of unit i makes on unit 0 (the detector), and it is updated according to

$$\begin{cases} \hat{W}_{0_i}^r(t) = W_{0i}^r(t-1) + C_i s_0(t) x_{ir}(t) \\ W_{0i}^r(t) = \hat{W}_{0i}^r(t) / \sum_{i'=1}^{n}\sum_{r'=1}^{m} \hat{W}_{0i'}^{r'}(t) \end{cases} \quad (13)$$

It suffices to set m to the maximum number of occurrences of symbols in a sequence. For example, to recognize S_1, m can be set to any number larger than or equal to 3. The choice of m, the number of terminals of each unit, limits the number of occurrences of the same symbol in a complex sequence. The learning rule is the same as in (7) except that all the m synapses of a unit are modified. Due to normalization in (13), the connection weight $W_{0i}^r(t)$ is set to $1/(mn)$ initially.

With this modification of the model, the above conclusions (number 1° through 4°) with simple sequence recognition are established similarly. In particular, the maximization principle applies:

5° After the detector has learned a complex sequence S_0, only presentation of S_0 induces the maximum activity on the detector unit.

$$x_{ir}(t) = \begin{cases} T, & \text{if } s_i(t) = 1, r = 1 \\ x_{i,r-1}(t-1) - 1, & \text{if } s_i(t) = 1, r > 1, x_{i,r-1}(t-1) < 0 \\ x_{ir}(t-1) - 1, & \text{if } s_i(t) = 0, x_{ir}(t-1) > 0, y_i(t) = 1, \\ x_{ir}(t-1), & \text{otherwise} \end{cases} \quad (11)$$

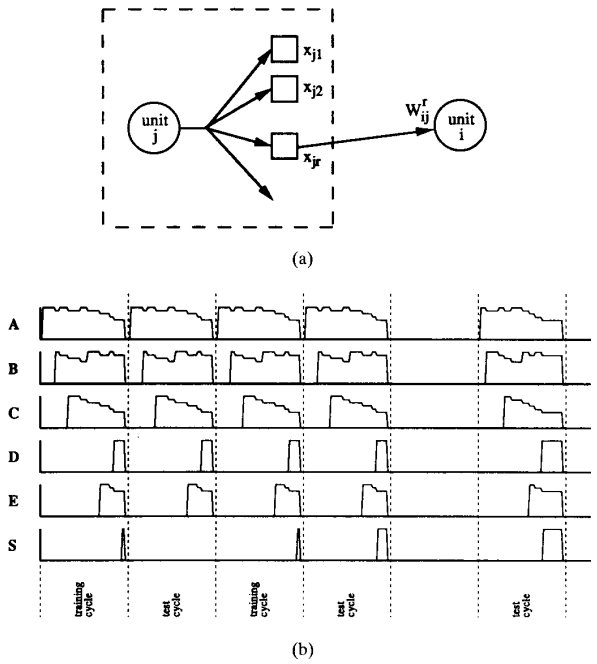


Fig. 3. (a) An expanded unit model for complex sequence recognition. A unit has multiple terminals that make contacts with other units. x_{j1} holds a trace of the most recent external input to unit j, x_{j2} the previous one, and so on, to a maximum of m such occurrences. (b) Recognition of the complex sequence S_1: *A-B-A-C-A-B-E-B-D* with time-warping. See the legend of Fig. 2(b) for understanding the plot. The generated presentation interval series was {9, 3, 6, 9, 5, 9, 7, 3, 6} for the sequence. After 6 training trials, unit 0 learned the sequence, i.e., could be activated by another presentation of the sequence. After that, the same sequence was again presented with the different interval series {4, 9, 4, 5, 8, 5, 4, 5, 3}, and the detector recognized the time-warped sequence. In the figure, only the last two training-test cycles are shown with the time-warping test. The parameters are $n = 10$, $m = 5$, $C_i = 0.02$ $(i = 1, \cdots, 5)$, $T = 10$.

The threshold Γ_0 in (12) can also be set similarly during the first training trial of the sequence. This conclusion guarantees that the model is able to recognize any complex sequence with time warp. As a demonstration, we simulated the above model for recognizing the sequence S_1. The attentional learning rule is used for complex sequence learning as before. During a training trial, each component had a presentation interval generated by a ranged random number generator. For S_1, the generated interval series was $\{9, 3, 6, 9, 5, 9, 7, 3, 6\}$. After the detector had learned the sequence, it was tested with another presentation of S_1 with interval series $\{4, 9, 4, 5, 8, 5, 4, 5, 3\}$ similarly generated. The detector correctly recognized the test sequence. Fig. 3(b) shows the simulation process for learning and recognizing sequence S_1.

The above model in principle can also handle, with a straightforward extension, temporal sequences that contain certain distortions in symbol form. By distortions in the symbol form of a sequence we refer to alterations of the sequence due to omissions or substitutions of its symbols, or additions of new symbols to it, not to distortions of the shape of an individual symbol. The latter should be handled during recognition of a specific symbol. We suggest to accommodate symbol form distortion by lowering the previous threshold value of the detector set by (10) a little so that the detector can also be triggered by a sequence similar in form to the learned sequence. According to (12), the detector measures the similarity between the learned sequence S_0 and an arbitrary sequence S_a by comparing the difference between the system-set threshold value IP_0 in (8) and the input potential (IP_a) induced by presentation of S_a. It does not appear straightforward to quantify the difference succinctly, a topic of future study. But we believe that such a quantification does exist in this model.

To shed some light on how the model measures the difference between S_a and S_0, let us assume $S_a = p_{a_1} - p_{a_2} - \cdots - p_{a_L}$. Repeated training with the complex sequence $S_0 : p_{0_1} - p_{0_2} - \cdots - p_{0_K}$ leads to a polarized weight distribution of the detector unit: $2(T-K+1)/[K(2T-K+1)], 2(T-K+2)/[K(2T-K+1)], \cdots, 2T/[K(2T-K+1)]$, and all others zero (see 1° above). Those linearly increasing weights correspond to $p_{0_1}, p_{0_2}, \cdots, p_{0_K}$ respectively, and the connections with these nonzero weights are called nonzero projections. Immediately after the presentation of S_a, the excitation levels of a set of units stimulated by S_a are $T-L+1, T-L+2, \cdots, T$. The similarity between S_a and S_0 depends on how many of the stimulated units by S_a can pass through a nonzero projection to contribute to IP_a, and how much those ordered units triggered by S_a match those triggered by S_0. As a particular example, when $S_a = S_0$, the weight vector of the detector after training with S_0 parallels the activity vector induced by presentation of S_a, and thus according to the Cauchy-Schwartz inequality IP_a reaches its maximum value that is equal to the threshold of the unit. In general, distortions towards the beginning of a sequence are less effective than those towards the end, because symbols towards the end have larger corresponding connection weights (see 1° above), and thus more leverage in determining sequence recognition.

There is always a tradeoff between tolerance and precision. After lowering of the thresholds of detectors, a specific sequence may trigger more than one detector, and a detector may be activated by more than one sequence, if we assume there are many detector units in the system for recognizing different sequences. Again a competitive network can help single out a detector that is trained by a sequence most similar to S_a.

One question not yet addressed with the recognition model concerns the length of a sequence that can be recognized. We have a hidden hypothesis when we develop the model, that is, the length of the sequence K should not be larger than the capacity T of the STM model. When the length of S_0 in question is larger than T, the above model only pays attention to and learns and recognizes the end subsequence $p_{0_{K-T+1}} - p_{0_{K-T+2}} - \cdots - p_{0_K}$ of S_0. As we know [44] human STM has a very limited capacity (7 ± 2), even though T can be set freely in engineering applications. But the above model is not sufficient as a cognitive model, since humans can memorize and recognize sequences much longer than ones directly limited by T. The next section addresses this problem.

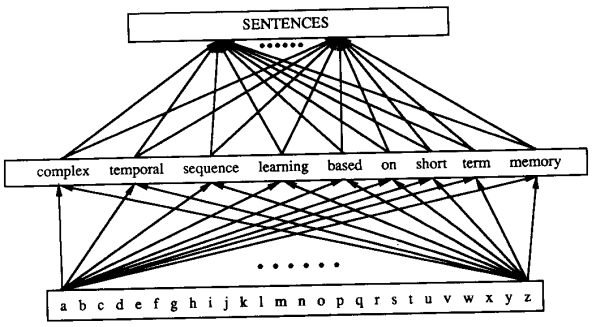


Fig. 4. Architecture of hierarchical sequence recognition. Different layers are connected in a feedforward manner from bottom to top. The bottom layer is the input layer, and others are detector layers of different levels. The letters and words symbolize different units in the layers.

IV. Hierarchical Sequence Recognition

Given the severe capacity limitations of STM, one method of reducing these limitations and so expanding our capacities is by chunking [44]. Sequence learning and sequential organization are obvious applications of the chunking notion. An example of chunking in sequence learning is the hierarchical organization of language. There is a series of hierarchies of sequential organization: the sequence of letters in a word, the sequence of words in a sentence, the sequence of sentences in a paragraph, the sequence of paragraphs in a discourse. Not only language, but all skilled actions seem to involve the same kind of hierarchical organization [40], [3].

Our hierarchical sequence recognition model, based on the chunking notion, consists of a cascade of layers of units. Units in layer i fully project to those in layer $i+1$, and each layer by itself is an STM model that is a fully connected network as shown in Fig. 1. The whole network is feedforward, and projections from a lower layer to a unit in the next higher layer are exactly like full projections of the units in the STM model to a sequence detector. This connection architecture is shown in Fig. 4 with an example sentence, S_2: "complex temporal sequence learning based on short term memory." Three layers shown in the figure are the letter layer, the word layer, and the sentence layer.[2] We use this example for demonstrating how chunking is done in the model, without suggesting that people process this specific sentence in the same way.

Let x_{ir}^l represent the excitation level of terminal r of unit i in layer l, and s_i^l the internal state of the unit. The weight of the connection from terminal r of unit j in layer l to unit i in layer $l+1$ is represented by W_{ij}^{lr}. The dynamics of x_{ir}^l and s_i^l are the same as before ((11) and (12)), and modification of W_{ij}^{lr} is also the same ((13)). The first layer directly interacts with the external environment. Therefore s_i^l is driven by external inputs. All other layers organize information from the basic input perceived by layer 1. All units in higher layers are sequence detectors, thus different layers detect different levels of information from an external input sequence. In Fig. 4, for instance, layer 1 detects individual letters, layer 2 detects individual words composed of sequences of letters, layer 3 detects individual sentences composed of sequences of words, and so on. The higher a layer is in the architecture, the higher is the level of input hierarchy that a unit in the layer can detect and the longer is an input sequence that the unit can recognize.

During training, units in higher layers are activated by the attentional learning rule. Previously, the attentional rule was only applied to a single detector, but now there are many detector units in a higher layer. Another attribute of the definition of attentional learning is that the internal activation of detectors must be sequential. This requirement is consistent with a basic property of attention. A unit in any higher layer has exactly the same model as a unit in the first layer. Therefore when attention shifts from a unit to another in a higher layer, activation thus triggered will drive the excitation levels of all other active units of the same layer to the next lower level due to mutual inhibitory connections. In other words, a higher layer forms its own STM due to sequential shifts of attentional activation. The STM model in a higher layer operates in the same way as one in the first layer, but with a larger time scale. So different time scales are formed automatically in different layers. This also explains why units in higher layers can recognize longer sequences.

How is attention allocated when there are different detectors in different layers? The correct order of attentional shift should be from lower to higher layers, because before layer i has been trained for recognition there is nothing to attend to for layer $i+1$. Taken Fig. 4 as an example, words in the sentence have to be attended to and learned before a unit in the sentence layer is attended to. As a result, a detector in a higher layer needs a longer time to learn. This is reasonable because the detector usually learns and recognizes a longer and more complicated sequence. To learn the sequence S_2, for example, the fastest possible way would have two stages. The first stage would present the sequence repeatedly until detectors in the word layer have learned each individual word. Suppose this stage takes X trials, depending on the value of C_i in (13). After this stage is finished, that is, presentation of the sequence alone can activate each word detector in the word layer, the next stage would be to learn the sentence as the ordered sequence of words. This stage takes another X training trials. So all together it would take at least $2X$.

A next question is when attention should be paid, or when a detector should be activated by the system. There is no general rule except that attention is activated at the end of presentation of a subsequence. In written English, for example, attention to words can be prompted by word separators like blank, comma etc., and attention to sentences can be prompted by sentence separators like period, semicolon, and so on. In speech, attention prompters could be sharp transitions between

[2] Hierarchical structure in language is, of course, more subtle than "crude chunking" since the "chunk" is based on syntax and semantics, rather than on a setting of some T. Thus, for example, a sentence is not represented directly as a string of words, but rather as a string of strings of strings ... corresponding to a syntactic/semantic parse tree for the sentence. In particular, the strict separation of levels adopted here must give way to a more flexible format that allows recursive specification of linguistic entities. It is beyond the scope of the present article to address such aspects, let alone the crosslinks for cross-reference that turn the tree into a more general graph structure. We want simply to note that our theory of chunking may play a crucial role in later studies of connectionist approaches to language. Language understanding is a very complex issue, and involves many other processes like long-term memory, communication and so on, but STM undoubtedly plays a critical role in it [7].

vocal movements, pauses between words, etc., though the difficulty of segmenting normal "running" speech poses problems, beyond those for "well-defined" sequences in written text or slow well-enunciated speech, that are not addressed here. Existence of these separators is not limited to language. Actually, it is because of the (sometimes implicit) existence of these separators that one can speak of the hierarchical organization of temporal sequences.

A computer simulation of the model is partly shown in Fig. 5 for recognizing sequence S_2. The first layer contains units for representing the basic symbols the 26 English letters. The second layer contains word detectors, among which the 10 words in S_2 are represented. The third layer contains sentence detectors, among which the sentence S_2 is represented. During training, the sequence was repeatedly presented, and attention was allocated according to the strategy described above. That is, words are attended to and learned first, and the sentence is attended to after the words have been learned. The model took 12 training trials to learn the sequence. Interval invariance is automatically achieved, because it is an intrinsic property of the recognition model defined previously. As in previous simulations, the interval of each component presentation was generated by a ranged random number generator, and a test was conducted using a different sequence of intervals after the model has learned S_2. Different time scales are clearly exhibited in the figure if letter units are compared with word units.

The capacity parameter T has been set to 10, yet the length of sequences that the model can recognize is not limited by T. In the above simulation, for example, the length of S_2 is 53. The length of sequences that a hierarchical model can learn and recognize increases exponentially with the number of layers in the model. Say T equals 10, the maximum length of a sequence learnable from units in the third layer is 100, from units in the fourth layer 1000, and so on. As noted before, from the engineering perspective, if we do not constrain the value of T long sequences can also be learned and recognized without resort to the hierarchical architecture. But it should also be noted that after the model learns S_2, it is able not only to recognize the whole sentence, but also to recognize individual words in the sentence independently and seems to involve a measure of parsing.

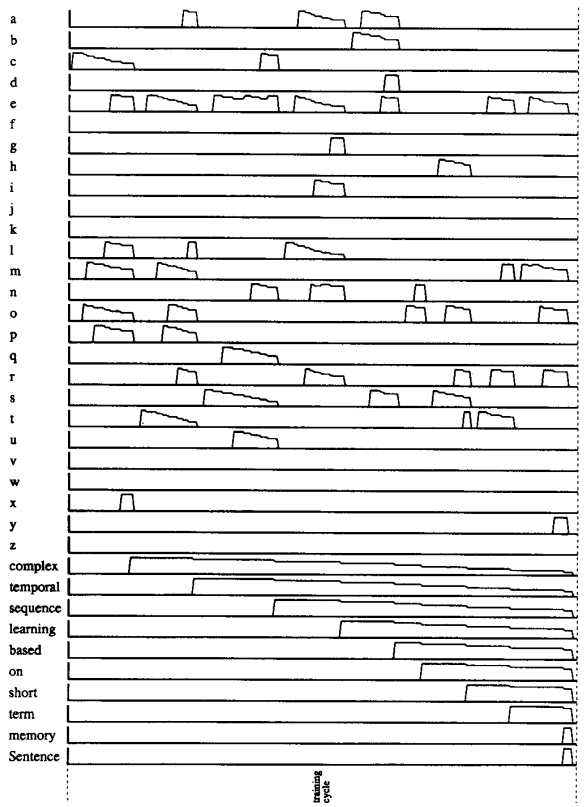


Fig. 5. Computer simulation of the hierarchical sequence learning model for recognizing S_2: "complex temporal sequence learning based on short term memory." Only one training cycle is shown in the figure for clarity. During training, the presentation interval series for S_2 is randomly generated. After 12 training trials, the model learns to recognize S_2. We only show the external levels of the units, without displaying their multiple terminals. For each letter unit, we see that it is activated to its maximum when its letter appears, then decrements as each subsequent letter is introduced, resetting to zero when the word separator is encountered. During training, a unit for a given word is activated at the end of presentation of that word (the attentional learning rule) and thus learns the sequential letter structure of that word on the basis of the letter units active at that time. Once the constituent words have been learned, the same mechanism can be applied one level further up the hierarchy to train the sentence unit to recognize the given sequential order of the words in S_2. In the simulation, all units have three terminals, $C_i = 0.3$ for all units in different layers, and $T = 10$.

V. Complex Sequence Reproduction

Sequence reproduction, or generation, is a different and somewhat more difficult task of temporal information processing than sequence recognition. As mentioned in the introduction, various solutions have been proposed for reproducing simple sequences, the main idea being to store transitions for each pair of consecutive patterns. Wang and Arbib [63] proposed a further model for complex sequence reproduction based on the learning mechanism for complex sequence recognition and the separation of detector units from symbol units. With this same scheme for dealing with complex sequences, the present model attempts to propose a solution for the time-warp problem with reproduction. Although interval invariance is desired for sequence recognition, sequence reproduction requires an opposite solution: *interval maintenance*. A dynamic tuning mechanism is also presented for degree self-organization of detector units.

The structure of the model for sequence reproduction has two layers, as shown in Fig. 6. Layer ζ is called the input layer, which basically serves as an STM model shown in Fig. 1. Multiple occurrences of a particular symbol in a sequence is represented by one single unit in this layer, so different units represent different spatial patterns in layer ζ. Units in layer ξ function as sequence detectors as described previously, and there is a global inhibitor within this layer (see Footnote 1). These units recognize the contexts of individual components in a sequence, and anticipate the occurrence of these components. Layer ξ connects with layer ζ bidirectionally, and before training connections between them are full. The projections

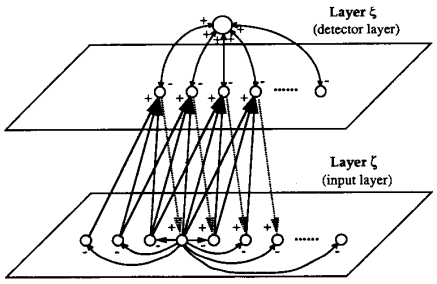


Fig. 6. Architecture for complex sequence reproduction. Within layer ζ (the input layer), every unit inhibits every other one to form an STM model as shown in Fig. 1. Within layer ξ (the detector layer), all units project to a global inhibitor that further projects back to them. At the beginning, the connections between layer ζ and layer ξ are all-to-all correspondence. The appropriate connection pattern between them for reproduction will emerge after repetitive training with temporal sequences. Plus signs indicate excitation, and minus signs indicate inhibition.

shown in Fig. 6 depict what results from training, so that unit i in layer ξ receives projections only from those units in ζ that represent symbols in the context detected by unit i, and unit j in layer ζ only receives input from units in ξ that anticipate the occurrence of the symbol represented by unit j. During the training process, a sequence with various component intervals is presented to layer ξ. At the end of each component presentation, a unit in layer ξ is randomly selected (but fixed in successive trainings[3]) to fire. That is, training of units in layer ξ follows the attentional learning rule. The recurrent connections from layer ξ to layer ζ are formed according to a Hebbian rule as follows. If unit i in layer ζ (recorded as $< i, \zeta >$) and unit j in layer $\xi (< j, \xi >)$ are firing simultaneously then a connection link from $< j, \xi >$ to $< i, \zeta >$ is established, and its weight is denoted as $W_{ij}^{\zeta\xi}$ which will be defined later. All connection weights from units in ξ to those in ζ are initially zero.

A. Degree Self-Organization

The global inhibitor in layer ξ receives input from all units in the layer and projects back to them. A degree parameter d_i is introduced for $< i, \xi >$, and it affects the dynamics of the internal state of $< i, \xi >$ in the following way (cf. (12))

$$s_i^{\xi}(t) = f(\sum_{j=1}^{n}\sum_{r=1}^{m} W_{ij}^{r} h(x_{jr}(t-1), d_i) + I_i^{\xi}(t-1) - \Gamma_i^{\xi}) \tag{14}$$

$$h(x,y) = \begin{cases} x & \text{if } x > T - y \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where label ξ in (14) indicates layer ξ, x_{jr} is the excitation level of the rth terminal of unit $< j, \zeta >$, and W_{ij}^{r} represents the connection weight from the rth terminal of $< j, \zeta >$ to $< i, \xi >$. Symbols n and m stand for the number of units and the number of terminals for each unit in layer ζ. The domain of d_i is $\{1, 2, \cdots, T\}$. Through function $h(x, y)$ the role of d_i is to gate in certain excitation levels of units in layer ξ. For instance, if $d_i = 1$, then only when x_{jr} equals T can unit $< j, \zeta >$ affect $< i, \xi >$. That is, if $< i, \xi >$ has degree 1, it can only sense the most recent item occurring in layer ζ. Obviously, the larger is d_i, the more items can $< i, \xi >$ sense from layer ξ. The formulations of W_{ij}^{r} and Γ_i^{ξ} are modified accordingly

$$\begin{cases} \hat{W}_{ij}^{r}(t) = W_{ij}^{r}(t-1) + C_i s_i^{\xi}(t) h(x_{jr}(t), d_i) \\ W_{ij}^{r}(t) = \hat{W}_{ij}^{r}(t) / \sum_{j'=1}^{n}\sum_{r'=1}^{m} \hat{W}_{ij'}^{r'}(t) \end{cases} \tag{16}$$

$$\Gamma_i^{\xi} = \frac{2}{d_i(2T - d_i + 1)} \sum_{i=1}^{d_i} (T - d_j + i)^2. \tag{17}$$

Let the activity of the global inhibitor of layer ξ be represented by z, and q represent the number of units in layer ξ. Variable z is defined as

$$z(t) = f(\sum_{i=1}^{q} s_i^{\xi}(t-1) - 2) \tag{18}$$

and therefore the inhibitor will be activated if there is more than one unit firing simultaneously in layer ξ. According to (14), the internal state $s_i^{\xi}(t)$ can be triggered either by system attention through $I_i^{\xi}(t)$ or by input signals from layer ζ. The latter is called *anticipation*. What the inhibitor actually does is to detect conflicts among those detectors in layer ξ. Since system attention is always sequential, the inhibitor can only be activated by conflicting attention and anticipation or just by conflicting anticipation from the detector layer.

Degree $d_i (i = 1, \cdots, q)$ is initially set to 1. Self organization of d_i is done according to

$$d_i(t) = d_i(t-1) + 1 \text{if } s_i^{\xi}(t-1) = 1, z(t) = 1, d_i(t-1) < T \tag{19}$$

that is, the degree of $< i, \xi >$ increments if this unit together with other units causes activation of the global inhibitor. If the degree of $< i, \xi >$ increments, there will be one more unit from the input layer that can be sensed by $< i, \xi >$. Thus the previously learned weight distribution to the unit (see (16)) will have to change its direction of distribution. In the situation, the model re-initiates the weight distribution of $< i, \xi >$ and threshold Γ_i^{ξ} is also modified according to (17) based on the new value of d_i. From (14), (15) and (16), it is clear that if $d_i(t)$ grew larger than T, the STM capacity of layer ξ, it would be equivalent to $d_i(t) = T$ in the dynamics of the internal state and weight distribution of $< i, \xi >$. That is why $d_i(t)$ has an upper limit of T. Value T consequently limits the degree of a sequence to be reproduced.

A computer simulation of the model was conducted for reproducing a complex sequence $S_3 : J - B - A - C - D - A - B - A - E - F - A - B - A - G - H - A - B - A - H - I$. Learning a complex sequence is slower than learning a simple sequence, because the complex sequence needs to dynamically increase the degrees of certain detectors, and each time such self organization is done earlier training of those detectors is discarded. Roughly speaking, time required for training

[3] This could be implemented by self-organization as demonstrated by Malsburg [43] and Kohonen [32]. For simplicity, the current system simply "remembers" the initial choice.

increases linearly with the degree of a sequence. It took 18 training trials before the model learned to reproduce S_3, whereas 6 trials sufficed to reproduce a simple sequence. Due to the training scheme, the number q must not be less than the length of the sequence minus 1. For S_3 of length 20, 19 units were selected in layer ξ and trained to anticipate the second to the last component of S_3 respectively. The degree vector acquired by the self-organization mechanism is $\{1,2,3,1,1,2,3,4,1,1,2,3,4,1,2,2,3,4,2\}$ for those detectors. The ninth component E, for example, must memorize the 4 prior components $D-A-B-A$ in order to be generated; the second component B, however, only needs to memorize the previous component J. In the sequence $A-B-C-A-B-D-A-B-E$, it might be argued that symbol B does not need to memorize 2 prior components, as produced by the above algorithm, but one prior component since symbol B is always preceded by A. However the result produced by the algorithm is justified if we generally allow each component being presented for a different interval. In this situation different A's preceding symbol B may have different time intervals in presentation, and therefore are, strictly speaking, different.

The above neural algorithm optimally identifies the amount of context required to reproduce any complex temporal sequence unambiguously. The context degree vector reveals many properties of the sequence being reproduced. For example, the degree vector produced with S_3 reflects, among other things, whether a component is preceded by a single component or by a recurring subsequence, and where a recurring subsequence starts and ends in the sequence. We believe that this kind of information is important in self-detection of recurring subsequences and generalization of a temporal structure from many sequences (like a grammar from sentences). These open issues are critically important for further studies of temporal order.

The same problem of finding the minimum amount of context has been studied by Kohonen [33] for producing unambiguous inference rules in sequence generation. The proposed solution, termed "dynamic expanding context," relies on explicit rules for resolving inference conflicts. The right hand side of an inference rule is a symbol in a sequence and the left hand side of the rule consists of the context of the symbol. All left hand sides are initially set to the predecessors of the right hand side symbols, and later repetitive scanning will expand left hand sides as necessary for resolving conflicts. All rules are stored in a table, and a significant amount of table searching is required by the system. The method has been applied to speech recognition and music generation [33], [34]. A basic difference of our proposal is that we do not resort to any external rules. Units representing symbols and detectors in our model are connected in a neuron-like manner, and communication among units is typically neural. Thus information is distributed over units and connections, and sequence processing is parallel. High-level operations, like table lookup or memory search, are avoided in the system. With little modification, our model should be able to apply to those application domains explored by Kohonen.

B. Interval Maintenance

In our model, the interval length of a component presentation is the time period during which the external input E of the unit corresponding to that component is equal to 1. This is equivalent to the period when the excitation level of the unit equals T. In the above model of sequence reproduction, a unit in layer ξ detects the onset of the context of a component in order to trigger that component in the reproduction process. For sequence S_3 above, for example, there is a detector in layer ξ that is trained to detect the context $D-A-B-A$ and to anticipate the onset of symbol E. According to the model, after training this detector is activated just one time step after the second A starts to occur (see (14)). But E should not be triggered until the whole interval of A occurrence has elapsed. The idea for interval maintenance is to code intervals by connection weights from the detector layer to the input layer. Since the backward projections from layer ξ to ζ provide many-to-one correspondence, an interval can be simply coded by a backward connection weight such that temporal integration of the entire interval is required to trigger the next component. Due to the introduction of backward projections from layer ξ to layer ζ, the previous internal state of unit i in layer ζ is now defined as (cf. (1))

$$s_i(t) = \begin{cases} 1 & \text{if } E_i(t)=1, E_i(t-1)=0 \\ f\left(\sum_{j=1}^{q} W_{ij}^{\zeta\xi} E_j^{\xi}(t)\right) & \text{otherwise} \end{cases} \tag{20}$$

where $E_j^{\xi}(t)$ is a cumulative activity of unit $< j,\xi >$. Suppose that during training, each presentation of a sequence has the same interval series, then E_j^{ξ} is defined as

$$E_j^{\xi}(t) = \begin{cases} \sum_{\tau=t_1}^{t} s_j^{\xi}(\tau), & \text{if } s_j^{\xi}(t)=1 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

where t_1 is the start of the period during which unit $< j,\xi >$ is consecutively activated till time t. Note that the temporal integration E_j^{ξ} is easy to compute locally and recursively. At the end of this consecutively active period $t_2, E_j^{\xi}(t) = t_2 - t_1$. Training of the backward projections is defined by the following Hebbian rule

$$W_{ij}^{\zeta\xi}(t) = \begin{cases} \frac{1}{E_j^{\xi}(t-1)} = \frac{1}{t_2-t_1}, & \text{if} E_j^{\xi}(t)>0, s_j^{\xi}(t-1)=1, \\ & s_i(t-1)=0, s_i(t)=1 \\ W_{ij}^{\zeta\xi}(t-1), & \text{if } E_i(t)=0. \end{cases} \tag{22}$$

The condition that $E_j^{\xi}(t) > 0, s_j^{\xi}(t-1)=1, s_i(t-1)=0$, and $s_i(t)=1$ holds iff the detector of unit $< j,\xi >$ precedes the onset of the next symbol represented by $< i,\zeta >$ in the sequence. This time instant is the same as t_2. In conclusion, the time interval of a symbol presentation is coded as the reciprocal of the corresponding connection weight.

In general, one interval series of presentation may be different from another one. In order to cope with this situation, instead of storing one interval directly in a weight, two parameters are stored in the connection, one is an average μ of different training intervals and another is a deviation σ^2 of training intervals. During reproduction of a sequence, a Gaussian number is generated based on μ and σ^2, which

has the same function as $t_2 - t_1$ in (22). Each generated interval will also modify μ and σ^2 like a presentation interval. Therefore, learning is a process of forming μ and σ^2. Let e_i represent the interval of the ith presentation of a symbol. Two factors are taken into consideration in forming μ and σ^2. First, each interval should contribute a certain amount. This is called the averaging factor. Secondly, a recent interval should have more impact than a remote one. This is called the recency factor. These two factors are embodied in the following learning rules.

$$\begin{cases} \mu_1 = e_1 \\ \mu_{k+1} = (1-\beta)\mu_k + \beta e_{k+1} \end{cases} \tag{23}$$

where β is the recency parameter ranging between 0 and 1, which ensures that, except the first interval, the most recent interval has a constant contribution regardless of the presentation history. Expanding the above formula, we have

$$\begin{aligned} \mu_k = {} & \beta e_k + \beta(1-\beta)e_{k-1} + \beta(1-\beta)^2 e_{k-2} + \cdots \\ & + \beta(1-\beta)^{k-2}e_2 + (1-\beta)^{k-1}e_1 \end{aligned} \tag{24}$$

where $\beta + \beta(1-\beta) + \beta(1-\beta)^2 + \cdots + \beta(1-\beta)^{k-2} + (1-\beta)^{k-1} = 1$, so that the definition of μ_k is still a type of averaging. From (24) it is clear how each interval contributes to the overall average, and the more recent an interval is, the greater its effect. If we view the above weighted formulation of μ_k as the average from samples $e_1, e_2, \cdots, e_k$ taken with frequencies $f_1 = (1-\beta)^{k-1}, f_2 = \beta(1-\beta)^{k-2}, \cdots, f_k = \beta$, respectively, we can define σ_k^2 by the formula

$$\begin{aligned} \sigma_k^2 &= \frac{k}{k-1}\sum_{i=1}^{k} f_i(e_i - \mu_k)^2 \\ &= \frac{k}{k-1}\Big(\sum_{i=1}^{k} f_i e_i^2 - \mu_k \sum_{i=1}^{k} f_i e_i\Big) \\ &= \frac{k}{k-1}\{(1-\beta)^{k-1}e_1^2 + \beta(1-\beta)^{k-2}e_2^2 + \cdots + \beta e_k^2 \\ &\quad - \mu_k[(1-\beta)^{k-1}e_1 + \beta(1-\beta)^{k-2}e_2 + \cdots + \beta e_k]\} \end{aligned} \tag{25}$$

and

$$\begin{aligned} \sigma_{k-1}^2 = {} & \frac{k-1}{k-2}\{(1-\beta)^{k-2}e_1^2 + \beta(1-\beta)^{k-3}e_2^2 + \cdots \\ & + \beta e_{k-1}^2 - \mu_{k-1}[(1-\beta)^{k-2}e_1 + \beta(1-\beta)^{k-3}e_2 \\ & + \cdots + \beta e_{k-1}]\} \end{aligned} \tag{26}$$

which yields the following recurrence learning rule for the deviation

$$\begin{cases} \sigma_1^2 = 0 \\ \sigma_k^2 = \frac{k(1-\beta)}{k-1}[\frac{k-2}{k-1}\sigma_{k-1}^2 + \beta(e_k - \mu_{k-1})^2] \end{cases} \tag{27}$$

so that $\sigma_k^2 = 0$, if $e_1 = \cdots = e_k$.

With the learning rule of (23) and (27), interval maintenance defined above is thus achieved. This model ability should again be attributed to separation of context detection in layer ξ and symbol presentation in layer ζ. Because of this separation, a unique link can be established from the detection layer to the input layer, and this link is able to carry interval information without confusion even when allowing complex sequences. A computer simulation of the model was conducted to reproduce the complex sequence S_3. During training, the interval of each symbol was initially generated by a ranged random number generator, but fixed in subsequent training trials for simplicity. As previously stated, the model took 18 training trials to learn the sequence. The number of trials is basically decided by the requirement of degree self-organization. Let us define the initial context of a sequence as the beginning subsequence required to uniquely determine the rest of the sequence. After learning, the entire sequence with various interval lengths was able to be reproduced by presentation of its initial context, subsequence J in this case. Fig. 7 presents the simulation result, which contains a temporal course of the last training trial together with the reproduction process. For parameters see the legend of the figure. Since in this simulation the speed of presentation is the same from one trial to another, the acquired deviation for every link interval is zero. Therefore the time course of the sequence is faithfully preserved in reproduction. Reproduction of a learned sequence does not have to start with the initial context. Presentation of any subsequence can reproduce the subsequent part of the sequence, as long as the subsequence can uniquely determine the following part of the sequence.

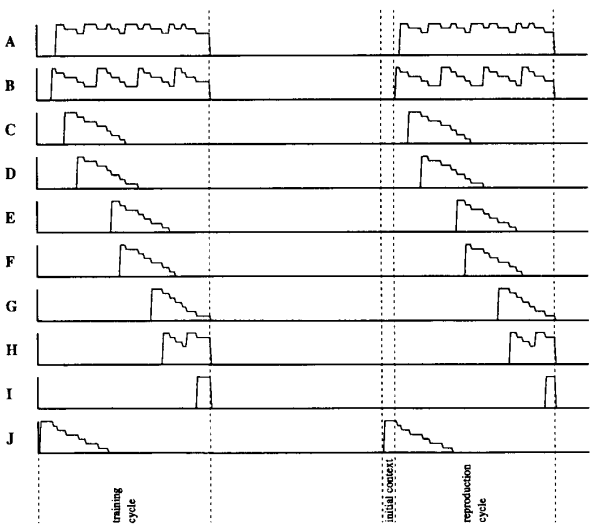


Fig. 7. Reproduction of the complex sequence S_3: *J-B-A-C-D-A-B-A-E-F-A-B-A-G-H-A-B-A-H-I*. The interval series {9, 3, 6, 9, 5, 9, 7, 3, 6, 4, 9, 4, 5, 8, 5, 4, 5, 3, 7, 8} was first randomly generated, and fixed in subsequent training trials. The model took 18 training trials before it was able to reproduce the entire sequence with presentation of S_3's initial context: J. Only the last training cycle and the reproduction cycle are plotted. Note that not only the order but also the time intervals of the sequence were reproduced. All units in layer ζ have three terminals, and $C_i = 0.3$. The other parameters are $\beta = 0.3$, and $T = 7$.

While the time course of a sequence can be reproduced by the model, the overall speed of reproduction can be easily controlled with a global rate tuning agent that projects to all synapses from layer ξ to layer ζ in the form of shunting inhibition. The agent can scale all $\mu's$ (averages) through shunting inhibition, thus implementing the "scaling effect" of

sequence generation. The scaling effect is often seen when musicians are learning a new piece: they practice it at a slow pace to get relative timing, and then play it faster and faster. It also occurs frequently in speech production [42]. A similar function has been recently demonstrated by Heskes and Gielen [25] based on a Hopfield-type associative memory model. The main idea was to use an adaptation scheme of neuronal thresholds, thus scaling delays between transitions of consecutive patterns in a sequence. Their method works well with a constant interval for all patterns, but it appears problematic to deal with variable intervals as handled in this paper.

VI. Discussion

A. On Complex Temporal Sequences

A basic feature of this model is to cope directly with complex temporal sequences, considering simple sequences as a specific case, whereas many other models do it the other way around. Complex sequences, in fact, are indispensable for almost every kind of natural temporal behavior, in reading, writing, speech production, music generation, skilled motor behaviors, and so on. Processing of complex sequences must be achieved before any neural network model can be applied to solving those problems of temporal order.

In the back propagation approach, when a state is fed back either from the output layer (the Jordan model [28]) or the hidden layer (the Elman model [18]), certain history information is preserved, and it was suggested that this feedback information can be used for disambiguation required in the complex sequence situation. In the Jordan model, the state that is used as input to generate the next component is coded as a temporal summation of a number of previous components in the sequence. Since the entire previous subsequence is coded by a single state, it is unwarranted that different subsequences can be uniquely recorded. Also, in order to let the same recurring symbol appear in the output layer, it is possible that dissimilar inputs[4] would have to yield the same output, while similar inputs would have to yield different outputs. This would make training very difficult, and result in poor generalization (see also [4]). In our model, however, a previous history is distributed among different units, each of which maintains its own activation over a variable amount of time, depending on further inputs to the STM model. Disambiguation thus can be ensured. The use of hierarchies has been suggested for helping reproduce a complex sequence in a backpropagation architecture, and this will be commented on later.

The Elman model has been mainly applied to recognize sequences with some formal description, like a subset of regular languages [18], [52]. Pollack also demonstrated a similar function using a different extension of the backpropagation network [48]. The difference between their work on temporal order learning and the kind of work presented here is that they are concerned with a set of sequences as a whole while we are concerned with learning (recognizing and reproducing) single sequences. These two types of study on temporal order are complementary, and both are necessary in a complete system of temporal order processing, say speech understanding, where recognizing single words is a necessary early step and recognizing grammatical structure is a later step.

In the approach based on spin-like neurons, recognition and reproduction of complex sequences have been studied by a number of authors. Tank and Hopfield [59] rely on a set of patterned delays to program the recognition circuit. However, no mechanisms have been proposed for how to acquire and maintain these delays. The selection model [11] uses high-order synapses, synaptic triads, for coding basic temporal order. To learn a sequence, high-order connections are randomly made and a desired architecture can be selected by an input sequence-simple or complex-according to the authors. Besides immense connections required to learn and reproduce a reasonably long sequence, it does not appear from the model that reproduction of an arbitrary complex sequence is guaranteed. The use of high-order synapses is also the key to the model by Guyon *et al.* [23], where the order of the synapse has to be made at least equal to the degree of the sequence in question. The system overhead due to the number of connections caused by introducing high-order synapses becomes a serious concern. A different method has been taken in the model by Kühn *et al.* [37] for complex sequence reproduction. They focus on one type of complex sequences, that is, sequences that contain only one recurring subsequence that itself is a simple sequence. A number of specific measures were taken to solve this kind of sequence generation problem, such as using two time scales for local and remote associations. This type of sequences belongs to the so called first-order complex sequences, which can be reproduced with a direct extension to the one layer network for sequence recognition [63]. To link remote components in a sequence, the present model does not resort to high-order synapses, but instead introduces multiple terminals for each unit. The number of connections used is thus a constant multiple (m) of those needed in a conventional network. In the high-order synapse scheme, on the other hand, the number of connections needed is D orders of magnitude higher, where D is the degree of the sequence.

In most of the neural network models for sequence processing, the time-warp problem has not been addressed, nor has generation of a sequence with different component intervals. Notably, Tank and Hopfield have addressed the problem by using broadly tuned time delays [59], an idea later extended to speech perception [60]. This scheme can successfully compensate limited variations of the presentation duration of each symbol. Since a delay range is attached to each symbol, rate invariance that requires a global constraint cannot be achieved. Anderson *et al.* [2] proposed a model for auditory pattern recognition based on a real-time recurrent learning rule [66]. They demonstrated that recognition is not affected if the presentation rate is reduced by half. The present theory provides a solution to the time-warp problem, in conjunction with the previous algorithm for complex sequence learning. In reproduction, in addition to general ability of interval

[4] Similarity here is measured by the Hamming distance, i.e., the number of different bits in two matrices.

maintenance, our system allows a different presentation speed of a sequence for each training and sequence reproduction does not generate a rigid time course, but rather is random within a certain range circumscribed by the recency and averaging factors. The ability to handle the time-warp problem and certain erroneous symbol problem in the domain of complex sequences represents a significant step forward in processing temporal order by the neural network approach.

B. Hierarchies

In continuation to the above discussion, hierarchies have been proposed as a way to cope with complex sequences [15], [27], [29]. A simple subsequence at some level is coded as a single symbol in the next higher level. During reproduction, components in higher levels are generated at slower time scales, allowing time for generating lower subsequences corresponding to these components. Within each level, only a simple sequence needs to be reproduced by a back propagation network. One obvious problem is that parsing of a sequence into different levels has to be provided externally by the designer in these networks, since back propagation networks have not been shown to be able to self-organize an elementary sequence into various hierarchies. As pointed out before, complex sequences are ubiquitous. If the basic network can only handle simple sequences, a great deal of parsing would be required before the network models can be used.

The idea of employing hierarchies is used differently in our model from their proposals. The motivation behind our proposal is to overcome the limitation of capacity of STM. Hierarchies are not required for processing complex sequences, since this is a basic capability of the model. For instance, the English word "efficiency"would require several hierarchies to be formed in the proposed back propagation models, and several ways exist to organize it into different hierarchies that contain only simple subsequences, and no solid reasons seem to favor one parsing scheme while rejecting others. This word would be naturally handled as a single entity in our model. Since STM capacity limits human temporal order processing and since chunking is the basic means for humans to organize temporal information, we therefore can largely rely on natural delimiters when we use the present model to hierarchically process long and complicated sequences arising from natural temporal behaviors.

Using performance measures, Nissen and Bullemer [46] have demonstrated that attention is required for subjects (humans) to learn to reproduce a temporal sequence of symbols. Under distraction with dual-task conditions, acquisition of the sequence was minimal. The sequence used in the investigation was $S_4 : D-B-C-A-C-B-D-C-B-A$, a complex one. A more detailed study was done recently by Cohen *et al.* [8] using the same experimental technique. They studied three example sequences that can be symbolized as $S_5 : A-E-B-D-C, S_6 : A-D-C-A-C-B$, and $S_7 : A-C-B-C-A-B$, and are classified as unique, hybrid and ambiguous sequences respectively. During training, each sequence repeats itself, thus forming continuous cycles. From the experimental results they conclude that the unique and hybrid sequences can be learned by subjects under attentional distraction, but the ambiguous sequence is much more difficult to acquire under the same attentional distraction. Interestingly, they suggest that ambiguous sequences involve hierarchical representation and thus require attention. From the present model, we would like to offer a different explanation of their data. More attention is required to learn and reproduce complex sequences (S_4, S_6 and S_7) than simple sequences (S_5) because degree self-organization is required in layer ξ when reproducing a complex sequence. The model further predicts that higher degree complex sequences are more difficult to acquire than lower degree complex sequences. For example, S_4 has degree 3 and would be more difficult to learn than S_6 which has degree 2. This is because higher degree sequences need more levels of self-organization according to the present model than lower degree ones. As for S_6 and S_7, although they have the same degree of 2, every symbol in S_7 has degree 2, whereas two symbols in S_6 have degree 1. Thus, S_6 is easier to learn than S_7. In sum, our explanation differs from theirs in that we see S_5 and S_6 belonging to different classes while they do not. In fact, their data to us favor a three-way classification more than a dichotomy. In addition, our explanation clears a confusion that seems to exist in their paper. That is, S_6 would need a hierarchical representation as well and thus attention if hierarchies are required for reproducing complex sequences, contradicting their conclusion that it can be acquired under attentional distraction.

It is interesting to notice linkage between the attentional learning rule and attentional requirements in learning sequences by humans. Attention is perhaps also needed for learning a simple sequence, like S_5 above. The difference revealed in acquiring simple sequences and complex sequences may suggest different amounts of attention required. Even under distraction with dual-task conditions, it is hard to say that attention is fully excluded in performing sequential tasks. In fact, from the revealed data curves [46], [8], there is a tendency to acquire complex sequences even under the dual-task distraction.

Although our demonstration of hierarchical representation of temporal sequences uses natural separators to form different levels, people also use other heuristics for chunking. For example, in the U.S., the 10-digit phone number 2137406991 is often parsed into three chunks: area code (213), then 3 digits (740), then 4 digits (6991). It is thus clear that a full theory of chunking will include a knowledge-dependent theory of "high-level"chunking to complement the purely sequence based "low-level" chunking studied here. Another related issue is how attention is allocated in the situation of hierarchical sequence recognition. We call this the scheduling problem. In the simulation of the hierarchical sequence recognition section, the scheduling simply followed the bottom-up strategy. That is, a sentence is attended (learned) only after its constituent words have been learned, and a word is attended only after its constituent letters have been learned. Scheduling will become more complicated if chunking of a sequence without explicit separators is required. Scheduling is related to selective attention, where a number of neural models have been developed (see among others [13], [31]). An obvious extension

to the present model is to have a competitive network model for self-organization of attention scheduling that can avoid having an external instructor teach the system.

C. Efficiency

Attentional training for sequence recognition or reproduction, depending on the value of the gain factor C_i ((6)), usually takes from several to tens of trials before the model learns the task. By tuning the value of the gain factor, the speed of learning can be controlled externally. Even in hierarchical sequence recognition and complex sequence reproduction, the most time-consuming tasks, the speed only deteriorates linearly with the number of layers or the degree of a sequence. The number of training trials needed for the model is comparable with that for humans in performing similar tasks [46], [8]. Not only that, the present model exhibits remarkable computational advantages over other models. In back propagation models, training a network usually takes thousands or more trials [50]. This amount of training cannot, of course, be avoided for the models that use the back-propagation algorithm for sequence reproduction. In spin-like network models, since associations are preprogrammed in the network, no training is involved in general. Even so, it usually takes a significant amount of time for the system to settle down to an equilibrium state of the dynamics.

As mentioned in Introduction, it has been proposed to use a backpropagation network with a buffer for temporal sequence recognition [62], [39]. Since a temporal sequence is converted into a spatial pattern, the proposal works for both simple and complex sequences. However, this leads to much more connections in the network that need to be adjusted by training, thus resulting in a formidable amount of training time. According to Waibel [61], 18 days on a 4-processor Alliant supermini was required to train the network to recognize 6 stop consonants ($/b, d, g, p, t, k/$). Even in terms of hardware use (space), our model also compares favorably with the buffer scheme. For complex sequence recognition, we introduce multiple terminals for each unit to hold multiple occurrences of a recurring symbol. In a sense, multiple terminals play the same role as a buffer except an important difference. Terminals in our scheme shift only when a symbol repeats in a sequence, whereas in the buffer scheme a shift of all "units" (so to speak) is required at every time step. Thus, our approach saves hardware use significantly. For example, 3 terminals are sufficient to learn sequence S_1 (see Fig. 3), but a buffer of length 9 is required for the buffer scheme.

D. Units

The building blocks of the present theory are units, which can be thought of as local neuron populations. Many functions of a unit, like spatial summation of inputs (see (12)), temporal summation of a single connection (used for interval maintenance), connectional plasticity (see (13)), etc., resemble those of a single biological neuron. Yet some more functions are assumed. The most outstanding one is, perhaps, introduction of multiple terminals for a single unit. These terminals could anatomically correspond to multiple neural fibres or many synaptic terminals possibly efferent from a neuron assembly. As discussed before, having different symbols represented by different units ("grandmother cells") is consistent with the concept of local neuron populations, which has been utilized in other situations [5], [64], [56]. Our results suggest an important computational function that could emerge from interactions among neuron assemblies. More distributed representations of the individual functional units should be possible.

We have not gone into detailed neural circuits for implementing units of local neuronal populations. A style of this implementation of local populations can be found in Buhmann and Schulten [5] and in Sporns *et al.* [56]. Neural oscillations might be able to provide an implementation of the excitation levels of units that only take discrete numbers [19], [64], [63], [56]. In this representation scheme, a neural oscillator that can be implement by a neuronal assembly (see [56]) would correspond to a unit, and the amplitude of an oscillation would correspond to the excitation level of a unit. In Wang and Arbib [63], discrete amplitudes are demonstrated by reciprocally connected units that damp autonomously in time, corresponding to the decay theory. This new model, however, would require replacement of the autonomous decay by the one driven externally by other units.

E. Studies of Delay Intervals,

The recency factor of interval acquisition in the learning rule of (23) and (27) is supported by biological data. Kojima and Goldman-Rakic [35] found that in performing delay tasks, a group of prefrontal neurons in monkeys displayed time-dependent firing patterns. In their experiments, delays of 2, 4 and 8 s were employed for training the monkeys to depress the hold keys until the delay period ended. By increasing the length of the delay, latency of firing activity and the position of firing peak were observed to readjust to the changes in the anticipated time of the delayed response. Readjustment was observed after about 10 trials with a new delay period. We see this readjustment as manifestation of the recency factor in forming a delay interval, although the qualitative nature of the data precludes a more detailed analysis.

Less direct evidence comes from studies of the interstimulus intervals (ISI) in classical conditioning. After repeated pairing of a conditioned stimulus (CS) and an unconditioned stimulus (US), the animal will develop the conditioned response (CR) after presentation of the CS, and the distribution of CR latencies centers near the ISI [54]. In the rabbit's nictitating membrane response, it has been observed that when ISI shifts from one value to another, the CR latency originally conditioned to the first value was found to shift rapidly to the one corresponding to the second value [54], [41], [9], [26]. This evidence is typical of the recency factor. In human eyelid conditioning, what Ebel and Prokasy [17] observed from the CR latency distribution well conforms to our learning rule. As training with a fixed ISI progresses, the standard deviation of the CR decreased, and both mean and standard deviation of latency varied directly with shifts in ISI.

More specifically, take one particular link in the above model as example. If the link is trained with interval $\triangle t_1$

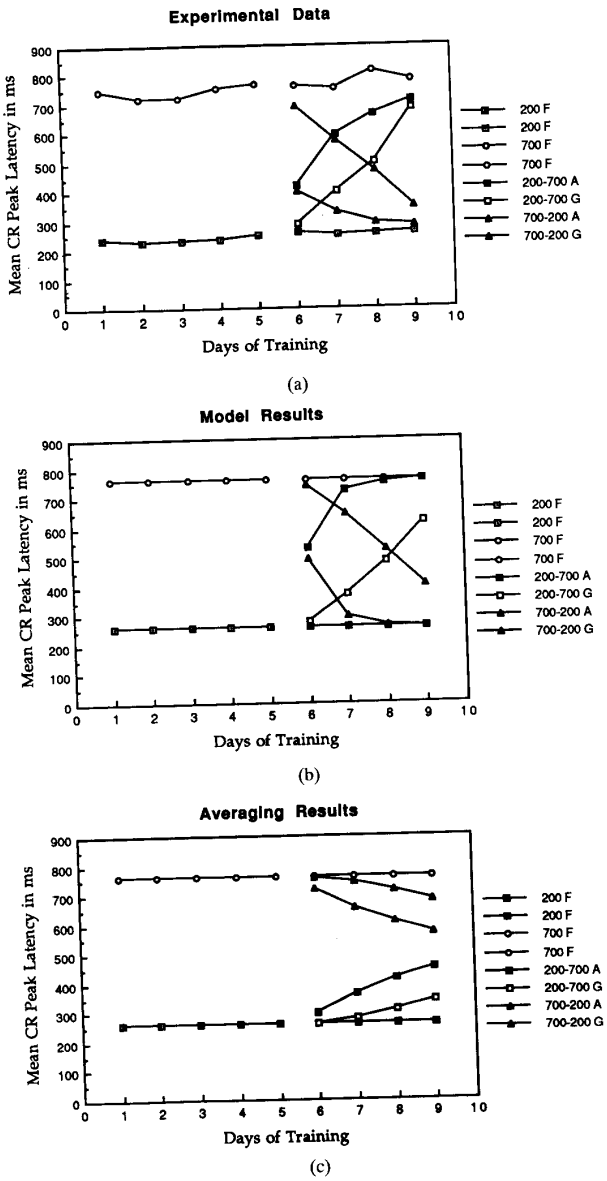


Fig. 8. Data and model outputs for shifts in mean CR peak latency in blocks of 9 test trials (days) for the 200- and 700-ms. In the plots, ISI conditions are indicated by the labels for all six groups, with F for fixed, A for abrupt, and G for gradual. Each of the nine days of training consisted of 90 paired CS-US trials and 10 CS-alone test trials, and beginning with the fifth trial of each 100-trial session, every tenth trial was a test trial. Each point represents the mean value of CR peak latency over 10 test trials within a day. Under the gradual condition, ISI incremented or decremented in steps of 25 ms after every 20th trial; and for the abrupt condition, ISI was immediately shifted from one interval to the other throughout days 6-9. (a) Experimental data. Each condition group contained 12 individuals, and the result was the average over the group (redrawn from [9]). (b) Model results. The same group of ISI conditions were used to yield the comparable results with the experimental preparation. In the simulation, $\beta = 0.02$. (c) Simulation results from the pure averaging method (see text).

first, and later it is trained with interval $\triangle t_2$, the above learning rule predicts that the interval acquired by the link will shift gradually from $\triangle t_1$ to $\triangle t_2$, due to the recency and averaging factors. This model phenomenon is strikingly similar to the data of Coleman and Gormezano [9] from classical conditioning of the rabbit's nictitating membrane response, as summarized in Fig. 8(a). They studied the effect of ISI shifts by employing ISIs of 200 and 700 ms and three subsequent ISI shift conditions (fixed, gradual, and abrupt) with two directions of ISI shift (short to long and long to short). In the gradual shift condition, the intermediate ISIs between 200 and 700 ms were used during the shift training (the last four days of a nine-day period), while in the abrupt case, the animal was first trained in days 1-5 with one interval, and then trained in days 6-9 with another interval. Our model response with the learning rule (23) was shown in Fig. 8(b). Compared to Fig. 8(a), the model yields not only comparable quantitative results, but also similar time courses. In particular, both the animal group and our model exhibit a linear shift for gradual conditions, and exponential shift under the abrupt condition. As a comparison, Fig. 8(c) shows the results produced with a pure average model, where $\mu_k = \frac{1}{k}\sum_{i=1}^{k} e_i$ (cf. (23)). It is clear to see, without the recency factor, how poor the results are. Furthermore, the learning rule (23) predicts that the amount of prior preparatory training does not affect the later shift in mean CR peak latency. More specifically, in the Coleman and Gormezano experiment, we predict that the same shift occurs even with just one day prior training (instead of five days in the original experiment). The similar gradual shift was observed in the CR topographies (instantaneous CR amplitudes) in the direction of the ISI shift. A later observation by Hoehler and Thompson [26] confirmed the systematic changes in the CR topographies in the direction of the ISI shift.

F. Cognitive Aspects

The present model is based on the interference theory of forgetting, retroactive interference in particular [65]. Our work demonstrates that a drastic difference in computational power could be gained by adopting a different view from basic studies of cognitive science. The computational model of STM offered here represents a simplified view, and has not incorporated other characteristics like proactive interference and the similarity factor. Nonetheless, our theory presents first attempts to solve complex problems using basic cognitive models.

Another cognitive source of the STM model is from Miller's work [44]. The magic number seven plus or minus two is explicitly incorporated into the excitation level of a basic unit (parameter T). The technique to overcome the capacity of the STM model, i.e., using hierarchical representation of temporal sequences, is directly inspired by the chunking idea that reveals how humans process information [44], [53]. Different layers in the model (see Fig. 4) correspond to different levels of the hierarchical representation, and process different extents of temporal sequences. The hierarchical model of sequence recognition suggests that there should be a distinct STM within each layer, and thus different levels of STM that function rather independently. Different time scales are characteristic of different levels of STM, but each STM obeys the same

description, like the capacity limit, interference and so on. This is a novel prediction of the theory for human behaviors of STM at the cognitive level. This prediction could be tested, for example, by allowing a subject to read or to listen to a piece of hierarchically organized material, and later asking what the subject can identify or recall from different levels of hierarchies. Of course, our model represents just a much simplified view on the richness of hierarchical knowledge representation and the chunking theory. Different levels of hierarchies may not correspond to different physical levels of neural networks, and a mechanism of establishing multiple levels of hierarchies within a single level of network may be desired.

How to recognize a long sequence, like one composed of hundreds, even thousands, of components? Two ways may be possible from the present theory. One is to utilize the hierarchical scheme described before. Because the number of elementary components in a sequence that can be recognized increases exponentially with the number of layers in the model, the hierarchical scheme offers a very effective method for recognizing long sequences. Yet another way is to make use of the process of sequence reproduction. Long sequences could be very simple (in terms of the degree of a sequence), and to reproduce them only needs to detect subsequences whose lengths are not larger than the degrees of the sequences, and may not involve recognition of long subsequences at all. In other words, the idea is to transform recognition of long sequences into reproduction that requires only recognition of possibly much shorter subsequences. The price of that would be an extra comparison of the sequence reproduced by the model (mentally) with the one being presented externally.

VII. Conclusion

The goal of this paper is to explore mechanisms for processing temporal order. A unified theory is provided for learning, recognition, and reproduction of complex temporal sequences. The entire model is built upon units corresponding to local neuron populations, and thus suggests a new level of modeling. Time intervals of sequence components do not affect recognition, but are preserved in reproduction. The present computational theory is inspired by cognitive studies not only in formation of the short-term memory model, but also in modeling of hierarchical information chunking. We hope that this computational theory can serve both as a theory of temporal order learning in humans and higher animals, and an effective algorithm for engineering applications of temporal information processing.

We demonstrate throughout the paper that complicated aspects of temporal order can be achieved by temporal linkage (local and remote) among different levels of sequence components, going much beyond what can be achieved by simple associative chaining as rejected by Lashley [40]. Meanwhile, we realize that many other problems with temporal order, like goal-directed planning, syntax formation, and hierarchy construction, remain largely untouched. However, we believe that the theoretical framework lays a sound ground for further study of temporal integration.

Acknowledgment

The authors wish to thank Chris Barker and two anonymous referees for their critical comments on earlier versions of this manuscript.

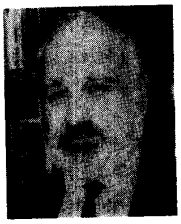
References

[1] S. Amari and M. A. Arbib, "Competition and cooperation in neural nets," in *Systems neuroscience*, J. Metzler, Ed. New York: Academic, 1977, pp. 119–165.
[2] S. Anderson, R. F. Port, and J. D. McAuley, "Dynamic memory: a model for auditory pattern recognition," preprint.
[3] M. A. Arbib, "Programs, schemas, and neural networks for control of hand movements: Beyond the RS framework," in *Attention and performance XIII. Motor representation and control*, M. Jeannerod, Ed. Hillsdale, NJ: Erlbaum, 1990, pp. 111–138.
[4] B. T. Bartell, G. W. Cottrell, and J. L. Elman, "The role of input and target similarity in assimilation," in *Proc. 13th Ann. Conf. Cog. Sci. Soc.* Hillsdale, NJ: Erlbaum, pp. 322–327, 1991.
[5] J. Buhmann and K. Schulten, "Noise-driven temporal association in neural networks," *Europhys. Lett.*, vol. 4, pp. 1205–1209, 1987.
[6] G. A. Carpenter, and S. Grossberg, "A massively parallel architecture for a self-organizing neural pattern recognition machine," *Computer Vision, Graphs, and Image Processing*, vol. 37, pp. 54–115, 1987.
[7] P. A. Carpenter and M. A. Just, "The role of working memory in language comprehension," in *Complex information processing: The impact of Herbert A. Simon*, D. Klahr and K. Kotovsky, Eds. Hillsdale, NJ: Erlbaum, 1989.
[8] A. Cohen, R. I. Ivry, and S. W. Keele, "Attention and structure in sequence learning," *J. Exp. Psychol*, vol. 16, pp. 17–30, 1990.
[9] S. R. Colman and I. Gormezano, "Classical conditioning of the rabbit's *(Oryctolagus cuniculus)* nictitating membrane response under symmetrical CS-US interval shifts," *J. Comp. Physiol. Psychol*, vol. 77, pp. 447–455, 1971.
[10] R. Conrad, "Decay theory of immediate memory," *Nature*, vol. 179, pp. 831–832, 1957.
[11] T. Dehaene, J. P. Changeux, and J. P. Nadal, "Neural networks that learn temporal sequences by selection," in *Proc. Nat. Acad. Sci. USA*, vol. 84, 1987, pp. 2727–2731.
[12] R. L. Didday, "The simulation and modeling of distributed information processing in the frog visual system," Ph.D dissertion, Stanford University, 1970.
[13] R. L. Didday and M. A. Arbib, "Eye movements and visual perception: 'Two visual systems' model," *Int. J. Man-Machine Stud.* vol. 7, pp. 547–569, 1975.
[14] K. Doya and S. Yoshizawa, "Adaptive neural oscillator using continuous-time back-propagation learning," *Neur. Netw.*, vol. 2, pp. 375–385, 1989.
[15] K. Doya and S. Yoshizawa, "Memorizing hierarchical temporal patterns in analog neuron networks," in *Proc. Int. Joint Conf. Neur. Netw.*, vol. 3, San Diego, CA, pp. 299–304, 1990.
[16] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis.* New York: Wiley, 1973.
[17] H. C. Ebel and W. F. Prokasy, "Classical eyelid conditioning as a function of sustained and shifted interstimulus intervals," *J. Exp. Psychol.*, vol. 65, pp. 52–58, 1963.
[18] J. L. Elman, "Finding structure in time," *Cog. Sci.,* vol. 14, pp. 179–211, 1990.
[19] W. J. Freeman, Y. Yao, and B. Burke, "Central pattern generating and recognizing in olfactory bulb: A correlation learning rule," *Neur. Netw.,* vol. 1, pp. 277–288, 1988.
[20] S. Grossberg, "Some networks that can learn, remember, and reproduce any number of complicated space-time patterns—Part I," *J. Math. Mechan,* vol. 19, pp. 53–91, 1969.
[21] 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.
[22] H. Gutfreund and M. Mezard, "Processing of temporal sequences in neural networks," *Phys. Rev. Lett.*, vol. 61, pp. 235–238, 1988.
[23] 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.
[24] D. O. Hebb, *The Organization of Behavior.* New York: Wiley, 1949.
[25] T. M. Heskes and S. Gielen, "Retrieval of pattern sequences at variable speeds in a neural network with delays," *Neur. Netw.,* vol. 5, pp. 145–152, 1992.

[26] F. K. Hoehler and R. F. Thompson, "Effect of the interstimulus (CS-UCS) interval on hippocampal unit activity during classical conditioning of the nictitating membrane response of the rabbit (Oryctolagus cuniculus)," *J. Comp. Physiol. Psychol.*, vol. 94, pp. 201–215, 1980.
[27] P. J. Jennings and S. W. Keele, "A computational model of attentional requirements in sequence learning," in *Proc. 12th Ann. Conf. Cog. Sci. Soc,*. Hillsdale, NJ: Erlbaum, 1990, pp. 876–883.
[28] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *Proc. 8th Ann. Conf. Cog. Sci. Soc.* Hillsdale, NJ: Erlbaum, 1986, pp. 531–546.
[29] M. I. Jordan, "Learning to articulate: Sequential networks and distal constraints," in *Attention and Performance XIII. Motor representation and control,* M. Jeannerod, Ed. Hillsdale, NJ: Erlbaum, 1990.
[30] D. Kleinfeld, "Sequential state generation by model neural networks," *Proc. Nat. Acad. Sci. USA,* vol. 83, 1986, pp. 9469–9473.
[31] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," *Human Neurobiol.,* vol. 4, pp. 219–227, 1985.
[32] T. Kohonen, "Self-organized formation of topologically correct feature maps," *Biol. Cybern.* vol. 43, pp. 59–69, 1982.
[33] T. Kohonen, "Dynamically expanding context, with application to the correction of symbol strings in the recognition of continuous speech," in *Proc. Int. Conf. Neur. Netw.,* vol. 2, San Diego, CA, 1987, pp. 3–9.
[34] T. Kohonen, "A self-learning musical grammar, or 'associative memory of the second kind'," in *Proc. Int. Joint Conf. Neur. Netw.,* vol. 1, Washington, DC, 1989, pp. 1–5.
[35] S. Kojima and P. S. Goldman-Rakic, "Delay-related activity of prefrontal neurons in rhesus monkeys performing delayed response," *Brain Res.,* vol. 248, pp. 43–49, 1982.
[36] B. Kosko, "Bidirectional associative memory," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-18, pp. 49–60, 1988.
[37] R. Kühn, J. L. van Hemmen, and U. Riedel, "Complex temporal association in neural networks," *J. Phys. A,* vol. 22, pp. 3123–3135, 1989.
[38] S. Kurogi, "A model of neural network for spatiotemporal pattern recognition," *Biol. Cybern.,* vol. 57, pp. 103–114, 1987.
[39] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neur. Netw.,* vol. 3, pp. 23–43, 1990.
[40] K. S. Lashley, "The problem of serial order in behavior," in *Cerebral mechanisms in behavior,* L. A. Jeffress, Ed. New York: Wiley, 1951, pp. 112–146.
[41] D. W. Leonard and J. Theios, "Effect of CS-US interval shift on classical conditioning of the nictitating membrane in the rabbit," *J. Comp. Physiol. Psychol.,* vol. 63, pp. 355–358, 1967.
[42] W. J. M. Levelt, *Speaking: From intention to articulation.* Cambridge, MA: MIT Press, 1989.
[43] C. V. D. Malsburg, "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik,* vol. 14, pp. 85–100, 1973.
[44] G. A. Miller, "The magical number seven, plus or minus two: some limits on our capacity for processing information," *Psychol. Rev.,* vol. 63, pp. 81–97, 1956.
[45] B. B. Murdock, Jr., "Serial-order effects in a distributed-memory model," in *Memory and learning: The Ebbinghaus centennial conference,* D. S. Gorfein and R. R. Hoffman, Eds. Hillsdale, NJ: Erlbaum, 1987, pp. 227–310.
[46] M. J. Nissen and P. Bullemer, "Attentional requirements of learning: Evidence from performance measures," *Cog. Psychol.,* vol. 19, pp. 1–32, 1987.
[47] L. R. Peterson and J. P. Peterson, "Short-term retention of individual verbal items," *J. Exp. Psychol.,* vol. 58, pp. 193–198, 1959.
[48] J. B. Pollack, "The induction of dynamic recognizers," *Machine Learning,* vol. 7, pp. 227–252, 1991.
[49] S. K. Reed, *Cognition: Theory and Applications.* Monterey, CA: Brooks/Cole, 1982.
[50] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing,* D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: The MIT Press, 1986, vol. 1, pp. 318–362.
[51] 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 MIT Press, 1986, vol. 1, pp. 151–193.
[52] D. Servan-Schreiber, A. Cleeremans, and J. L. McClelland, "Learning sequential structure in simple recurrent networks," in *Advances in Neural Information Processing Systems 1,* D. S. Touretzky, Ed. San Mateo, CA: Morgan Kaufmann, 1989, pp. 643–652.
[53] H. A. Simon, "How big is a chunk?" *Science,* vol. 183, pp. 482–488, 1974.
[54] M. C. Smith, "CS-US interval and US intensity in classical conditioning of the rabbit's nictitating membrane response," *J. Comp. Physiol. Psychol.,* vol. 65, pp. 679–687, 1965.
[55] H. Sompolinsky and I. Kanter, "Temporal association in asymmetric neural networks," *Phys. Rev. Lett.,* vol. 57, pp. 2861–2864, 1986.
[56] O. Sporns, G. Tononi, and G. M. Edelman, "Modeling perceptual grouping and figure-ground segregation by means of active re-entrant connections," *Proc. Nat. Acad. Sci. USA,* vol. 88, pp. 129–133, 1991.
[57] J. C. Stanley and W. L. Kilmer, "A wave model of temporal sequence learning," *Int. J. Man-Machine Stud.,* vol. 7, pp. 397–412, 1975.
[58] S. Sternberg, "High-speed scanning in human memory," *Science,* vol. 153, pp. 652–654, 1966.
[59] 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.
[60] K. P. Unnikrishnan, J. J. Hopfield, and D. W. Tank, "Speaker-independent digit recognition using a neural network with time-delayed connections," *Neur. Comp.,* vol. 4, pp. 108–119, 1992.
[61] A. Waibel, "Modular construction of time-delay neural networks for speech recognition," *Neur. Comp.,* vol. 1, pp. 39–46, 1989.
[62] A. Waibel, T. Hanazawa, G. E. Hinton, K. Shikano, And K. J. Lang, "Phoneme recognition using time-delay neural networks," *IEEE Trans. Acoust. Speech and Signal Processing,* vol. ASSP-37, pp. 328–339, 1989.
[63] D. L. Wang and M. A. Arbib, "Complex temporal sequence learning based on short-term memory," *Proc. IEEE ,* vol. 78, pp. 1536–1543, 1990.
[64] D. L. Wang, J. Buhmann, and C.v.d. Malsburg, "Pattern segmentation in associative memory," *Neur. Comp.,* vol. 2, pp. 94–106, 1990.
[65] N. C. Waugh and D. A. Norman, "Primary memory," *Psychol. Rev.,* vol. 72, pp. 89–104, 1965.
[66] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neur. Comp.,* vol. 1, pp. 270–280, 1989.

DeLiang Wang was born in Anhui, the People's Republic of China in 1963. He received the B.Sc. degree in 1983 and the M.Sc. degree in 1986 from Beijing University, Beijing, China, and the Ph.D. degree in 1991 from the University of Southern California, Los Angeles, CA, all in computer science.

From July 1986 to December 1988 he was with Institute of Computing Technology, Academia Sinica, Beijing. He is currently an Assistant Professor of Department of Computer and Information Science, and Center for Cognitive Science at the Ohio State University, Columbus, OH. His present research interests include temporal pattern processing, auditory and visual pattern perception, neural mechanisms of visuomotor coordination, neural network theories, and computational neuroscience.



Michael A. Arbib was born in England in 1940 and grew up in Australia. He received the B.Sc. (Hons.) degree from Sydney University in 1961, and the Ph.D. degree in mathematics from the Massachusetts Institute of Technology, Cambridge, MA, 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 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 and Physiology, as well as of Biomedical Engineering, Electrical Engineering, and Psychology at the University of Southern California, Los Angeles, CA. The thrust of his current work focuses on the mechanisms underlying the coordination of perception and action, centered around the modeling of neural mechanisms of visuomotor coordination in frog and toad, and on mechanisms for eye–hand coordination in humans and monkeys.