Anticipation-Based Temporal Sequences Learning in Hierarchical Structure

Janusz A. Starzyk, *Senior Member, IEEE*, and Haibo He, *Member, IEEE*

***Abstract*—Temporal sequence learning is one of the most critical components for human intelligence. In this paper, a novel hierarchical structure for complex temporal sequence learning is proposed. Hierarchical organization, a prediction mechanism, and one-shot learning characterize the model. In the lowest level of the hierarchy, we use a modified Hebbian learning mechanism for pattern recognition. Our model employs both active 0 and active 1 sensory inputs. A winner-take-all (WTA) mechanism is used to select active neurons that become the input for sequence learning at higher hierarchical levels. Prediction is an essential element of our temporal sequence learning model. By correct prediction, the machine indicates it knows the current sequence and does not require additional learning. When the prediction is incorrect, one-shot learning is executed and the machine learns the new input sequence as soon as the sequence is completed. A four-level hierarchical structure that isolates letters, words, sentences, and strophes is used in this paper to illustrate the model.**

***Index Terms*—Hierarchical structure, input anticipation, temporal sequence learning, winner-take-all (WTA).**

I. Introduction

TEMPORAL sequence learning is presumably among the most important components of human intelligence. Over the past decade, models and mechanisms for temporal sequence learning have attracted considerable attention. Sun and Giles [1] offer a useful review of the characteristics, problems, and challenges for sequence learning from recognition and prediction to sequential decision making.

In [2], a dual-weight neural network (DNN) scheme for fast learning, recognition, and reproduction of temporal sequences was developed. In a DNN, each neuron is linked to other neurons by long-term excitatory weights and short-term inhibitory weights. Fast learning is achieved by employing a two-pass training rule to encode the temporal distance between two arbitrary pattern occurrences. Based on this, the DNN was extended to a more generalized model, DNN2, which incorporates a self-organizing algorithm. In this way, the DNN2 can achieve autonomous temporal sequence recognition and reproduction. In [3], an online sequential extreme learning machine (OS-ELM) was proposed for fast and accurate sequential learning. This algorithm can learn data either one-by-one or chunk-by-chunk. In [4], Tijsseling proposed a categorization-and-learning-module (CALM) network with time-delayed connections for sequential information processing. In the proposed CALM, the modules can autonomously adjust its structure according to the complexity of the application domain.

Time is critical for human intelligence and manifests itself in all kinds of sequential behavior, including reasoning, planning, goal creation, and goal driven learning. As Elman pointed out [5], the difficult issue is how to represent time in a natural and biologically plausible way. Traditional way of handling this problem is to explicitly represent time as an additional dimension together with the pattern vector. However, there is no evidence that biological systems store timing information this way. Elman presented several other drawbacks of traditional approach to handle sequence timing, and suggested that time should be represented implicitly by the effect of the sequence itself. Our model uses this natural approach to sequence timing.

Recurrent neural network (RNN) is a powerful tool for sequence learning and prediction [6], [7]. Characterized with the recurrent connection, RNN is able to memorize the past information, therefore, it can learn and predict dynamic properties of the sequential behavior. Applications of the hierarchical sequence learning in RNN are investigated by Tani and Nolfi [8], [9]. In [8], a hierarchical forwarding forward model (FF-model) based on RNN is proposed to learn and generate complex sequential behaviors. This model has a higher level and lower level RNN, and uses both bottom-up (recall the past behavior) and top–down (predict future behavior) interactions. Simulation results on an arm robot show that this model can learn and generate sequential behavior, and adapt to the environment. Hierarchical representation and learning in RNN for sensory-motor systems are investigated in [9], where multiple-module RNNs are used to learn and compete in predicting the sensory-motor flow. Simulation results on robot-based navigation learning problems show that this mixture of RNN experts architecture is able to learn the sensory-motor flow dynamically, and also is able to self-organize its structure across multiple levels for learning and prediction. However, a traditional RNN network with a backpropagation through time (BPTT) has some drawbacks as discussed by Hochreiter and Schmidhuber in [10]. One of the most important issues is that in the traditional gradient-descent algorithms for RNN, the error signals through the backpropagation can vanish [10]. This will prevent the RNN from learning and predicting longer time lag sequences. Based on this, an important contribution was made in [10] by proposing a long short-term memory (LSTM). LSTM is a modified RNN that is able to handle such a drawback in the conventional RNN. The key idea is that the LSTM features a memory cell based on the constant error carousel (CEC) that is able to sustain its activation over a

Manuscript received November 10, 2005; revised July 24, 2006; accepted August 8, 2006.

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

H. He is with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: hhe@stevens.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNN.2006.884681

long period of time, therefore, the error signal will not vanish and the system will be able to learn the sequential dynamics in very long time. Details about the architecture and applications of LSTM can be found in [10].

Wang and Arbib [11] proposed a complex temporal sequence learning model based on short-term memory (STM). In their paper, two special types of neurons are proposed for complex temporal sequence learning. The first one is the dual neuron, which is used to store a signal for a short period of time. Unlike the traditional binary signal values used in many neural network models, the output of a dual neuron is a graded signal. The second neuron is the sequence-detecting neuron. After learning, this sequence-detecting neuron fires in response to the previous sequence of patterns, not just the previous pattern. It thereby overcomes a limitation of networks that cannot reliably recall sequences that share common patterns.

The same authors presented a framework of learning, recognition, and reproduction of complex temporal sequences in [12]. In this model, sequences are acquired by the attention learning rule, which combines Hebbian learning and a normalization rule with sequential system activation. Time intervals between sequence components do not affect recognition. A global inhibitor was proposed to enable the model to learn context lengths required to disambiguate associations in complex sequence reproduction. In order to overcome the capacity limitation of STM, a hierarchical sequence recognition model based on the chunking notion was proposed in this paper. For instance, in a letter–word–sentence hierarchical structure, a unit for a given word is activated at the end of the presentation of that word, and the model learns the sequence of letters of that word on the basis of the letter units active at that time. Once the word structure has been learned, the same mechanism can be applied to train a higher hierarchical level on the sequence of words. One more issue addressed in this paper is interval maintenance. This is achieved by coding intervals by connection weights from the detector layer to the input layer.

In [13], a neural network model capable of learning and generating complex temporal patterns by self-organization was proposed. The model actively regenerates the next component in a sequence and compares the anticipated component with the next input. A mismatch between what the model anticipates and actual input triggers one-shot learning. Although the anticipation mechanism improves the learning efficiency of this model, it needs several training sweeps to learn a sequence. As the number of training sweeps required depends on the degree of complexity of the sequences, it is not very efficient for highly complex sequences.

Another paper [14] focused on the learning of the multiple temporal sequences and "catastrophic interference." It is showed that the anticipation model previously proposed by the same authors is capable of incremental learning with retroactive interference but without catastrophic interference. In addition, a chunking mechanism was included in this model to detect repeated subsequences between and within sequences, thereby substantially reducing the amount of retraining in sequential training.

Wang proposed the use of associative memory to learn and retrieve spatio-temporal sequences in [15] and [16]. In [15], the static associative neural networks with delayed feedback connections are used for learning and predicting the spatio-temporal sequences. In the learning stage, the primary input sequence is presented to the primary input channel, while the corresponding expected output (pairing sequence) is simultaneously presented to the pairing input channel. Learning of such spatio-temporal sequence is achieved by the heteroassociative memory networks (HANNs). After learning, the system can retrieve (predict) the whole spatio-temporal sequence from a small cue sequence presented into the primary input channel. Since the model in [15] assumes that each HANN only learns heteroassociations between single patterns and does not learn to associate multiple patterns in groups, the same authors extended the model [16] to include associations of one pattern with multiple patterns. The proposed model has the advantages of short learning time, fast and accurate retrievals, and the ability to store a large number of complex sequences consisting of nonorthogonal spatial patterns. In both of the models, the required number of networks (HANNs in [15] and multiassociative neural networks (MANNs) in [16]) for learning and retrieving a sequence increases with the complexity of the sequence. This is needed to handle complex spatio-temporal sequences. Recently, besides Wang's work, Chartier and Boukadoum proposed a sequential dynamic heteroassociative memory architecture that is capable of learning and retrieving spatio-temporal sequences [17]. Simulation results on gray-level images show the effectiveness of the proposed model. In addition, an associative memory model with a shift register structure for online temporal sequence learning, recognition, and prediction was proposed in [18]

A hierarchical structure is important for sequence learning. For instance, in [19], a linear-time algorithm called SEQUITUR is proposed for identifying hierarchical structure in sequences. The main idea of this algorithm is that phrases which appear more than once can be replaced by a grammatical rule that generates the phrase, and that this process can be continued recursively producing a hierarchical representation of the original sequences. Although this paper is not focused on sequence retrieval, it presents the idea of hierarchical representation of sequences, which is used in many papers that model sequence learning. For instance, George and Hawkins [20] discuss the problem of hierarchical structure for temporal sequence learning targeting invariant pattern recognition. This paper concludes that the neocortex solves the invariance problem in a hierarchical structure. Each region in the hierarchy learns and recalls sequences of inputs, and temporal sequences at each level of the hierarchy become the spatial inputs to the next higher regions. Assuming Markovian sequences at each level, the optimal Bayesian inference rule for any level in the cortical hierarchy is presented in this paper.

In our paper, we aim to develop a novel, biologically plausible temporal sequence learning model. Our model is hierarchically organized and efficient in both learning time and storage capacity. At each hierarchical level, a winner-take-all (WTA) structure is used to select the firing neurons of this level, and the firing neurons in one level provide the input to the next hierarchical level. A prediction mechanism is proposed in this model for efficient learning. As long as all predictions are correct, no learning takes place. Otherwise, a learning signal is issued at the end of a sequence. Since the input sequence is stored in input registers (IRs), one-shot learning is possible and no repetition of the training data is required. In our model, timing information is represented by the dynamics of sequential behavior established through sensory-motor coordination. In addition, we did not use any shift-registers to shift the sequence. Instead, time-pointer neurons together with a multiplexer (MUX)-based selection are used for the prediction mechanism.

Our approach bears some resemblance to the incremental learning of sequences presented by Wang and Yuwono [13], [14]. Our model differs, first, in the use of a hierarchical approach that extends the concept of learning by chunking. We believe that ours is a more natural approach to sequence learning and allows a natural grouping of sequences within the context of learned events.

A second difference relates to the efficiency of learning. In [13] and [14], training requires several presentations of a complex sequence that includes repetitions of the same subsequence in different contexts. (For simple sequences in which no subsequence is repeated, only one training sweep is necessary.) The training phase is completed when there is no mismatch occurring during the last training sweep. In our approach, a complex sequence needs to be presented only once. The number of retraining sweeps required in [13] and [14] is a function of the sequence degree. The upper bound of the sweep number is $k * (k + 1)/2$, where k is the sequence degree defined as the shortest prior subsequence that uniquely defines the following sequence component. This upper bound is reduced to one in our approach without sacrificing computational efficiency or increasing the hardware cost.

The third difference is the hardware implementation structure. In our approach, we use a memory-like storage of temporal events rather than the shift register structure proposed in [13] and [14]. Instead of shifting all inputs as each new element of the sequence is presented to the system, we store each new element at a specified location. Accordingly, the hardware cost in Wang's work depends on the degree of complexity of the sequence while in our method it depends on the sequence length. One can argue that the sequence degree is smaller than its length. However, the opposite is true for complex sequences hierarchically represented as sequences of elements representing chunked sequences. Take, for example, a sequence in which the same word is repeated more than once. The sequence degree is then at least equal to the length of this word, and if several words are repeated, to the combined length of these words. Chunking may help this to some degree, but it introduces a requirement for additional training to accommodate chunks. We note that this is not a natural approach adopted in biological neural systems. People are not capable of memorizing long sequences even if their order is low. Instead of learning a sentence as a long sequence of letters in which individual words are just part of the sequence, people learn sentences as sequences of words, learn stanzas as sequences of sentences, learn songs as sequences of stanzas, and learn stories as sequences of scenes. We elaborate on these differences in Section III.

What we have tried to preserve from Wang's and Yuwono's approach is its incremental learning of sentences that allows learning based on acquired knowledge rather than learning of all sequences at once. Incremental learning is natural for biological systems, where knowledge is built upon prior experience, modifies behavior, and influences further learning. Not all methods for learning temporal sequences preserve this important feature. For instance, the approach taken in [2] claims many advantages over the method presented in [14]. However, it is not an incremental learning model as it requires two-pass training. In the first pass, all temporal weights are computed and they affect the global threshold established in the second pass. Thus, whenever there is a change in the learning sequence, the whole training process must be repeated.

The importance of incremental learning and the difficulty involved in obtaining such learning has been known at least since Grossberg [21]. Subsequently, McCloskey and Cohen [22] identified the problem as the "catastrophic interference." Sharkey [23] characterized the catastrophic interference as one of forgetting previously learned information as new information is added. There have been many attempts to solve the problem by minimizing the overlap between synaptic weights storing various sequences [24]–[26] with little success demonstrated for relatively small sequences. Catastrophic interference in incremental learning is chiefly the result of modification of weights representing all previously stored patterns. While some interference may be found in human memory [27], the catastrophic interference is not. In our approach, ideally there is no interference between stored patterns, as a learning increment involves only those neurons not previously involved in information storage. Only when the number of presented sequences exceeds the storage capacity of our associative memory does partial interference appear. The catastrophic interference never appears.

Another important feature of Wang and Yuwono's approach preserved in our method is anticipation-based learning in which the model actively anticipates the next input. As long as the anticipated inputs are correctly verified, no learning is required on any level of the hierarchy. When a mismatch is detected on a given level of the hierarchy, a new sequence on this and all higher levels must be learned.

In addition, our model uses one-shot, normalized, Hebbian learning such that the entire sequence is learned using only a single presentation of the input data. Any new sequence at any level of the hierarchy is learned by self-organization that allocates the required resources. The proposed approach can store arbitrarily complex sequences as long as the number of distinct subsequences is smaller than the storage capacity expressed by the total number of neurons on various hierarchy levels.

Hierarchical sequence recognition is also considered in [28] and [29], where complex sequences are separated into simple sequences combined at higher hierarchical levels. In [12], hierarchical learning is used and requires a number of presentations of the training sequence before a higher level sequence can be learned. In our paper, learning on all levels proceeds without repeated input presentations, and sequence reproduction does not require special handling. In particular, there is no need for the interval maintenance required in [12].

The rest of this paper is organized as follows. In Section II, the overall architecture of our model is presented. A modified Hebbian learning mechanism and the detailed structure of each module is described. In Section III, simulation results of the proposed model for a four-level sequence learning problem

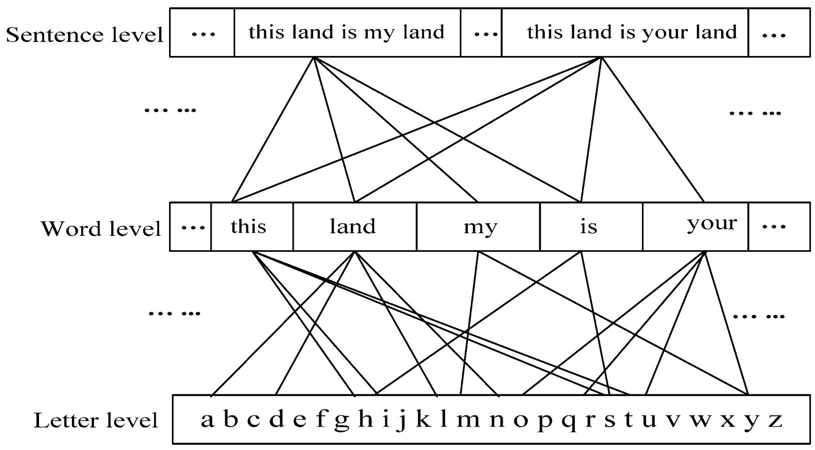


Fig. 1. Hierarchical organization of sequences.

are shown. This application divided a popular song into four hierarchical levels: letters, words, sentences, and strophes, and our model shows correct learning and retrieval of the complex sequences. Finally, a discussion and conclusion is given in Section IV.

II. System Architecture

In this paper, we focus on the learning and retrieval of multiple complex temporal sequences in the network of neurons. We adopt the terminology introduced by Wang and Arbib in [11] and [12]. A temporal sequence is defined as

$$S\colon S_1 - S_2 \cdots - S_n \tag{1}$$

where $S_i, i = 1, \ldots, n$ is a component of sequence S and the length of the sequence S is n. If a sequence includes repetitions of the same subsequence in different contexts, it is called a complex sequence, otherwise, it is a simple sequence. Fig. 1 shows the hierarchical organization to three levels of a letter–word–sentence. It is believed that such a hierarchical structure of a sequence is efficient [12], [19], [20].

The overall system architecture of our proposed model is shown in Fig. 2. It contains a hierarchical layer structure. The output of one hierarchical level is the input to the next level. At each level, a WTA algorithm is used to select active neurons.

Winning neurons are drawn in gray. In the first layer (level 0), a modified Hebbian learning mechanism is utilized for pattern recognition. Levels 1 to N are identical structures for temporal memory. The key components of each hierarchical level are: IR, MUX, prediction neuron (PN), prediction checking neuron (PCN), prediction matching neuron (PMN), learning flag neuron (LFN), multiple winner detection neuron (MWDN), and learning neuron (LN). As described later, the IR spatially encodes a temporal sequence of outputs from the next lower level. The sequence may be recent or recalled. The MUX calls the contents of the IR sequentially for comparison with a sequence being output by the next lower level. We will show that this network is efficient for complex temporal sequence learning and retrieval.

A. Pattern-Recognition Layer: A Modified Hebbian Learning Mechanism

A modified Hebbian learning mechanism is used in the first hierarchical layer of our model (level 0 on Fig. 2). Since biological neurons either fire or not, we assume that each sensory input from the environment is either 0 or 1, and use a both active-0 and active-1 representation of the sensory information. Fig. 3 illustrates this idea. When the sensory input is 1, the left neuron will fire. When the sensory input is 0, this input value is passed through an “inverter” and drives the right neuron to fire. In this way, different neurons firing represent different sensory input values for the binary coded input.

Fig. 4 shows the detailed structure of the modified Hebbian learning mechanism used in this paper. For simplification, we show a three level hierarchical structure with unsupervised learning. The neurons in the second layer are grouped into several groups. Each neuron of a second layer group is sparsely connected, at random, to the same subset of neurons in the first layer. There is some overlap of the subsets of neurons in the first layer to which second layer groups project.

Two similar WTA mechanisms are used in this paper to improve the learning efficiency of the model as well as to reduce its learning complexity. The first one is a stiff WTA (SWTA), which can be achieved by a simple counter. SWTA is used in the second layer. Since the sensory inputs from the environment are either 0 or 1, SWTA simply counts the number of 1s each neuron receives, and selects, as winners, the one neuron from each group that receives the largest number of 1s. The second WTA mechanism is used in the output layer (layer 3 in Fig. 4). Initially, all the weights w_i for neurons in the output layer are randomly set with the following conditions:

$$\begin{cases} w_{\text{ni}} = \pm 1 \\ \sum_i w_{\text{ni}} = 0 \end{cases} \tag{2}$$

where n indexes the neurons, and $i = 1, 2 \ldots k$ indexes connections onto neuron n from all the neurons of the previous layer. The winner is given by

$$\text{Winner } (w) = \max_n \left(\sum_i w_{\text{ni}} I_{\text{ni}} \right) \tag{3}$$

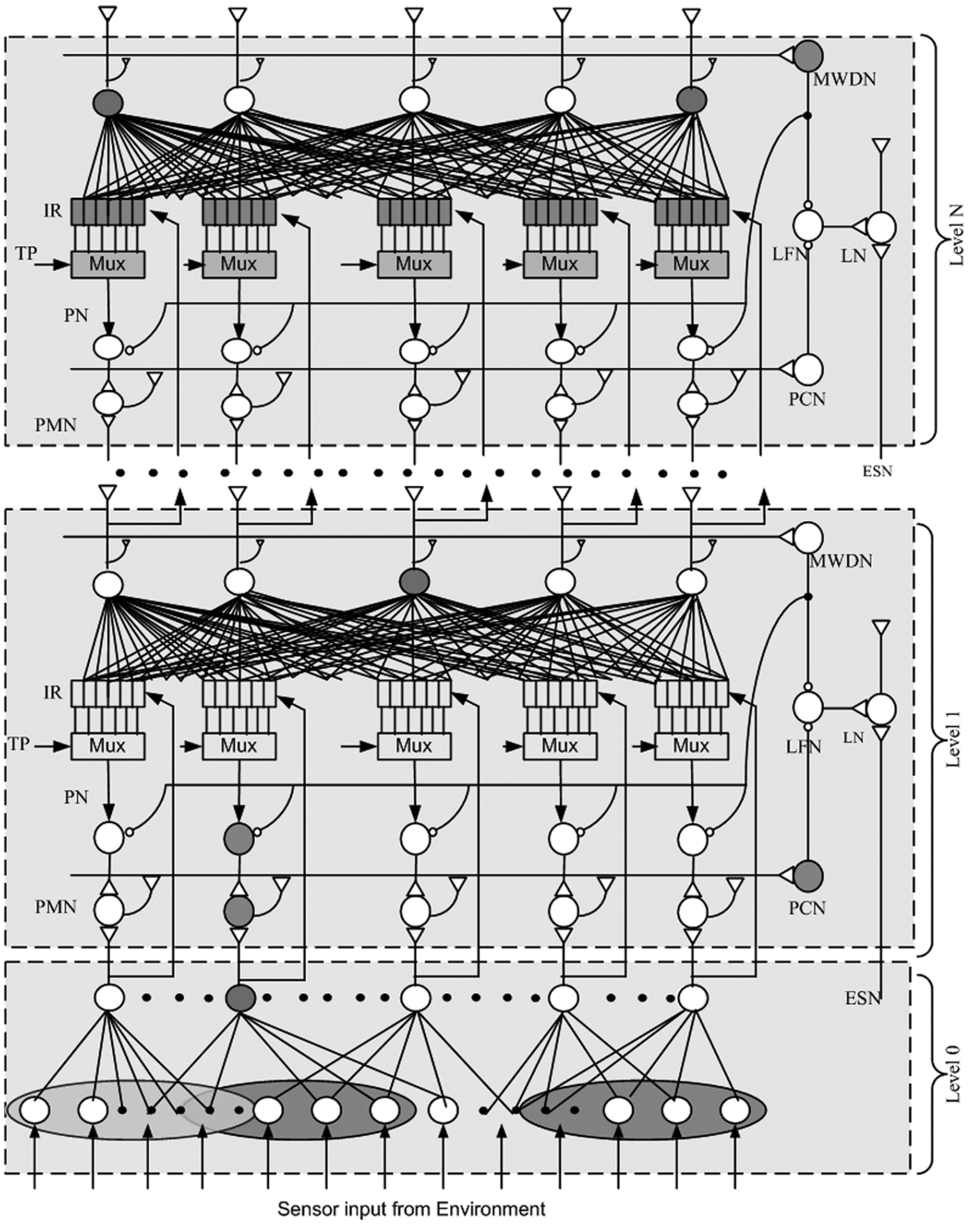


Fig. 2. Overall system architecture of the proposed anticipation-based hierarchical temporal sequence learning.

where I_{ni} is the activity (0 or 1) of the neuron from which projection w_{ni} arises. In each time instance, after the winner is selected, the weights of the winner (w_{wi}, where the subscript w denotes a winner) are adjusted as follows.

1) For the connections that received input $I_{\mathrm{li}} = 1$

$$w_{\mathrm{wi}}(t+1) = w_{\mathrm{wi}}(t) + \Delta * M(0). \tag{4}$$

2) For the connections that received input $I_{\mathrm{li}} = 0$

$$w_{\mathrm{wi}}(t+1) = w_{\mathrm{wi}}(t) - \Delta * M(1) \tag{5}$$

where Δ is a small adjustment, and $M(1)$ and $M(0)$ are the number of 1s and 0s the neuron received.

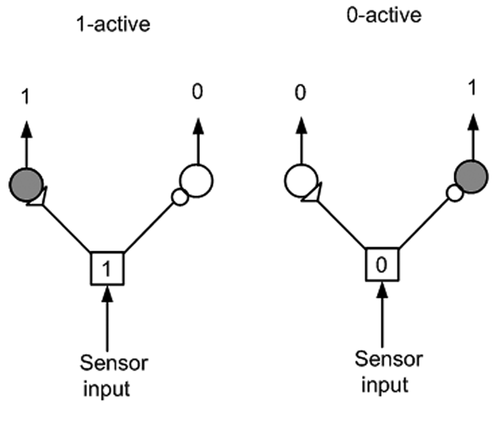


Fig. 3. Two-active area neuron firing mechanism.

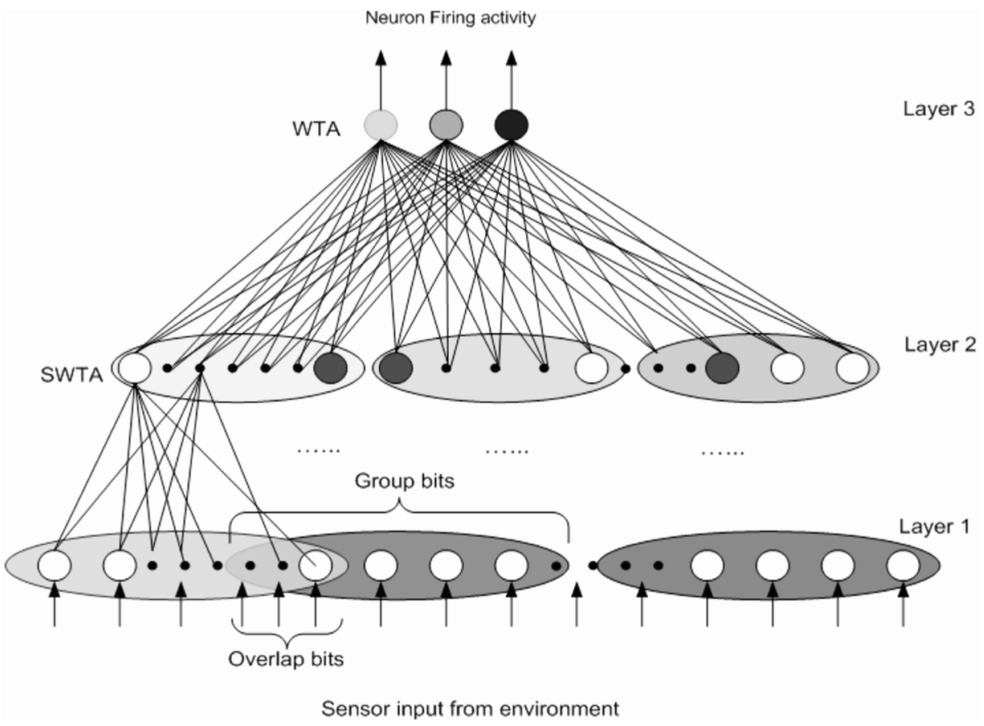


Fig. 4. Hierarchical structure of the modified Hebbian learning.

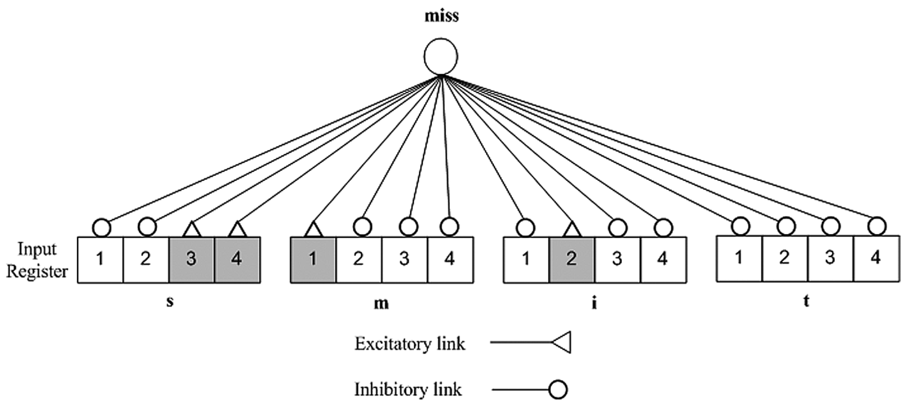


Fig. 5. Storage of sequence in IR.

This adjustment guarantees that the sum of the weights for the winner after adjustment still equals 0. After adjusting all the weights of the winners, we linearly scale their weights to be in the range $[-1, 1]$. Unsupervised learning is used in our simulation, meaning that each output neuron updates its activity whenever a new training sample is presented.

B. Storing Input Sequences

As indicated in the previous section, the first hierarchical level provides recognition of components $S_i, i = 1 \ldots n$ in the sequence S. From now on, we focus on how the model can store and predict complex sequences.

The output of one level is stored in the corresponding IR of the next higher level as shown in Fig. 2. Neurons of the IRs project onto the output neurons of the same level through trainable connections. Initially, all output neurons are fully connected to all IRs through untrained links with electrical (resistive) synapses. In our simulation, we randomly set the initial weights for all the output neurons to be a small positive number $0.001 < w_i < 0.01$.

Once an input sequence is stored in the IRs, all the output neurons compete and weights of the winning neuron are adjusted. All the weights of projections of active neurons onto a winning neuron in IR are set to 1 (excitatory) and all other weights onto a winning neuron are set to -100 (inhibitory). We employ this strong inhibition to guarantee that once an output neuron is trained to store a sequence, it is excluded from further learning.

Example 1: Fig. 5 shows the IR states for one neuron after training. The locations of the various IRs establish that this neuron stores the letter sequence "miss" (i.e., above the letter "m" there is a excitation link from the section of the IR neurons representing the first time step). Once an output neuron's links are trained, the neuron responds only to one specific input sequence.

Before discussing the prediction mechanism, we describe the structure of the IR. In our model, the IR stores the input data

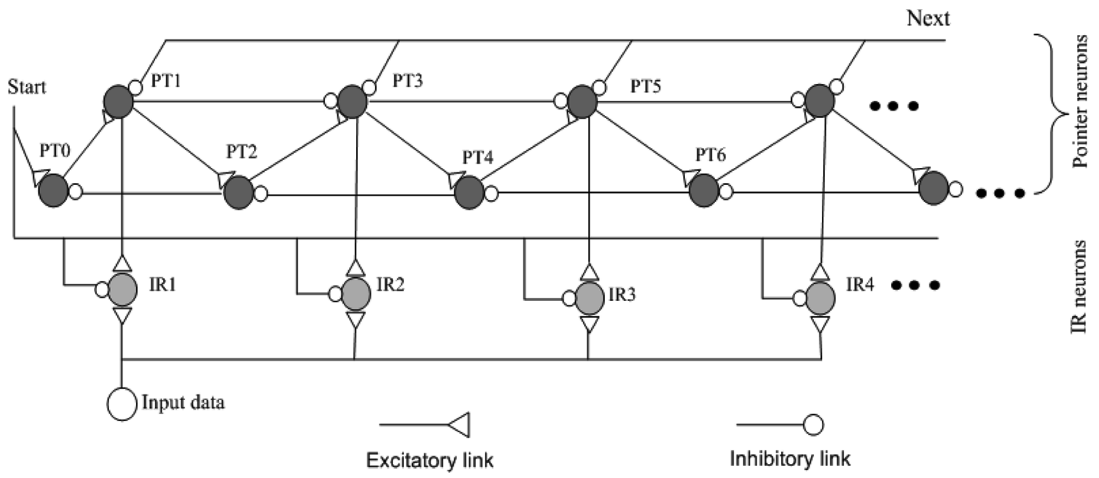


Fig. 6. Structure of IR.

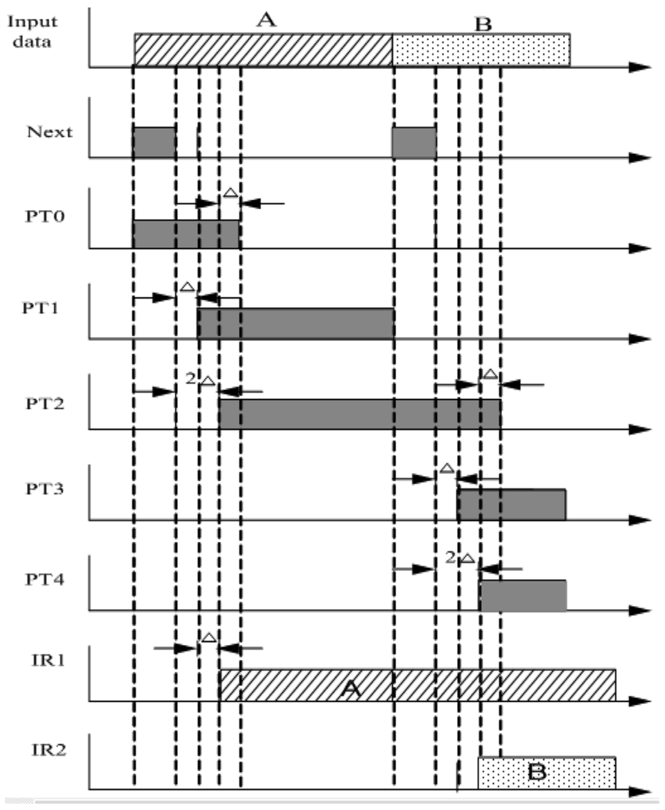


Fig. 7. Timing diagram of neurons firing in the IR.

selected by WTA in the location indicated by a time pointer. Each pattern in a sequence is directed to a different location. The structure of the IR is shown in Fig. 6.

Proposition 1: Given sufficient numbers of pointer neurons and IR neurons, the structure in Fig. 6 stores any input data by firing the corresponding neurons. This mechanism is biologically plausible since no computational weights adjustment is needed.

Proof: We consider a neuron firing delay Δ in activation of the links between neurons in Fig. 6. Consider also that we want to store a sequence S_n, where n is the sequence length. At the beginning of the new sequence, an impulse start signal is presented. The start signal clears all information in the IRs through inhibitory links. At the same time, PT0 fires. When the first data S_1 is received, the next signal is generated. When the next signal goes low, the inhibition of the upper pointer neurons is removed. Therefore, after the delay time Δ, PT1 will fire. After one more delay time, the lower pointer neuron PT2 fires since it has an excitatory link from PT1. At the same time, PT1 provides the inhibition to PT3. Suppose that at that time input data neuron was activated. This will happen if an input pattern, which corresponds to symbol S_1, (represented by this particular IR) was recognized and selected for storage. Since start signal is now low, and both PT1 and input data neuron are active, IR neuron IR1 fires, which stores the first data in the IR1. Assume that after some time, the second data in the sequence is presented and it is also the same symbol S_1. As before, the next signal inhibits all the upper pointer neurons PT1, PT3, PT5, and so on. However, PT2 continues firing. When the next signal goes low, then after the delay time Δ, PT3 fires because it is excited by PT2. As before, PT4 fires after PT3 with additional delay Δ. At the same time, IR2 fires provided that it is excited by PT3 and the input data neuron. This will store the data S_1 in the second input register IR2. If, however, signal S_2 was presented, then IR2 will be activated in the IR that stores S_2 signals. The process continues until the last data in the sequence S_n is presented.

As one can see from the previous analysis, the lower level pointer neurons provide inhibitory feedback to remove excitation from the prior pointer neurons and to excite the next pointer neuron. This approach forms long-term memories (LTM) of the learned sequences. One can make these memories modifiable by allowing training to decay after some specified period of time.

Example 2: Suppose we want to store the data "AB" in the IR neuron as illustrated in Fig. 6. Following the mechanism as described in Proposition 1, Fig. 7 shows the neuron firing diagram to store this input data. As we can see from Fig. 7, data "A" and "B" are correctly stored in the corresponding IR neuron and such storage will remain active until a new sequence is presented (i.e., the start signal will generate an impulse and clear all the previously stored data.)

C. Predicting the Input Sequences

1) Prediction Mechanism: Predicting an input sequence is an essential part of sequence learning in this model. By correctly predicting a sequence, the machine indicates it knows the sequence and does not require additional learning. If there is a single error, the LTM is modified to learn the new sequence. A simple one-shot learning mechanism is employed.

The first stage of prediction is a competition stage. Several sequences stored in the LTM compete to determine a unique sequence that may correspond to the input sequence. Once such a sequence is determined, it predicts the consecutive inputs. (Notice that the same mechanism is used if LTM is simply playing back a stored sequence after a high-level node of LTM is triggered by an internal process.) An MWDN is used to detect whether there are multiple winners with trained links. This is achieved by setting the threshold of the MWDN equal to 2. Therefore, it will fire when there are two or more winners in the output layer. (This occurs when the sum of the weights of all the winners are the same.) The output of the MWDN is connected to the LFN as well as all the PNs through inhibitory links. This provides a mechanism for setting the proper learning flag signal. The output of the LFN is connected to the LN through an excitatory link. Together with the excitatory links from the end of the input sequence neuron (ESN), the whole system provides a proper mechanism for setting the learning signal. It should be noted that each hierarchy level requires its ESN to indicate the end of input sequence for this level. In addition, if an ESN neuron fires in a higher hierarchical level, it automatically generates an ESN signal in all the lower hierarchical levels. (This signal flow is not shown in Fig. 2.) Consider an example in which the first level of hierarchy in LTM stores letters, the second level stores words, and the third level stores sentences. The last letter of every word will trigger the activation of the ESN neuron for the second level. The last letter of the last word in every sentence will be followed by the activation of the ESN for the third level, and at the same time, the ESN signal will also be sent to the lower levels (level 2 in this case).

Since there are three possible outcomes for the competition stage of the prediction mechanism, we now prove that under these conditions, the proposed architecture, as shown in Fig. 2, provides an efficient mechanism for sequence learning and prediction. Some details of the actual competition are left for subsequent sections.

Proposition 2: If there is a single winner with trained links in the competition stage, the architecture will either activate the PCN (meaning a correct prediction was made and no learning is needed so far), or activate the LFN (meaning a one-shot learning is needed for this new sequence).

Proof: In this case, only a single winner is firing through competition and MWDN does not fire because it has a threshold of 2. Therefore, the PN and the LFN are not inhibited. The set of PNs includes one PN corresponding to each output neuron of the lower level. By a mechanism described in Section II-C3, the LTM activates a PN that represents the network prediction for this time step. The time pointer (TP) is incremented with each new symbol (pattern) presented to LTM. As may be seen in Fig. 2, each PN receives an inhibitory projection from the MWDN, as well as the excitatory projection from the MUX. Since, in this case, the PN is not inhibited by the MWDN, LTM (acting through the MUX) activates a particular PN corresponding to the predicted symbol for the input sequence. Each PN and its corresponding output neuron from the next lower level form two inputs to a PMN. Firing of a PMN verifies that the predicted symbol corresponds to the input symbol. The PMN is activated only if the corresponding prediction and input neurons fire, as shown in Fig. 8.

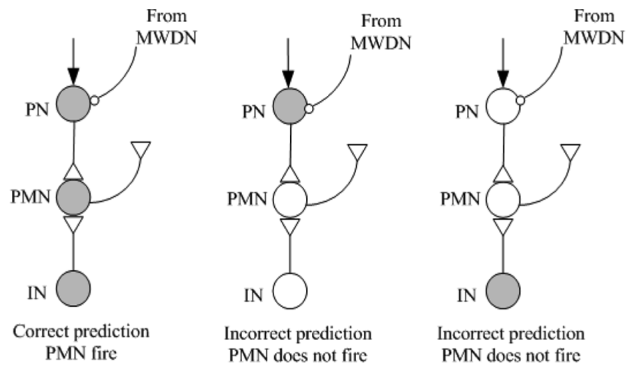

From MWDN
PN
PMN
IN
Correct prediction PMN fire
From MWDN
PN
PMN
IN
Incorrect prediction PMN does not fire
From MWDN
PN
PMN
IN
Incorrect prediction PMN does not fire

Fig. 8. Firing mechanism of PMN.

All PMNs have their outputs connected to the PCN, as shown in Fig. 9. This neuron fires to indicate a correct prediction.

If there is no match, the LFN is set automatically (no inhibition from the PCN or the MWDN). LFN remains on, and the sequence continues until the ESN fires. Thus, the firing of both LFN and ESN triggers the LN, as shown in Fig. 10(a). If there is a match on the output of the PMN, PCN will fire, and LFN is inhibited. Fig. 10(b) and (c) shows the remaining two conditions for activation of the LN. Fig. 10(b) indicates that if only the LFN fires (meaning there is no correct prediction), the LN neuron will not fire because ESN does not fire in this situation. Fig. 10(c) indicates that if only ESN fires, the LN will not fire because LFN does not fire (meaning there is a correct prediction).

Proposition 3: If there are multiple winners with trained links in the competition stage, no prediction is made.

Proof: Assume there are $n (n \geqslant 2)$ output neurons all have trained links to a particular IR neuron. Therefore, when that IR neuron is activated, all these n neurons will fire. Since the threshold of the MWDN is set to 2, therefore, MWDN will fire. From Fig. 2, we can see that MWDN will inhibit both PN and LFN. MWDN will prohibit the system from predicting what will be the next data. We will use an example to show this situation clearly.

Example 3: Consider the situation shown in Fig. 11 where two words "miss" and "mom" are already stored in LTM (only the excitation links are shown in Fig. 11) and the new sequence is "mit." When the first symbol of this sequence "m" is input, both neuron n1 and n3 win with trained links to "m" (weight equal to 1). MWDN then reaches its threshold of 2 and fires. As indicated in Fig. 11, MWDN inhibits both PN and LFN. Since both neurons are the winners with trained links, it is premature for the network to attempt a prediction. (Inhibition of all the PNs also reduces energy consumption.) When the second symbol "i" is presented, neuron n1 wins because it receives two excitatory

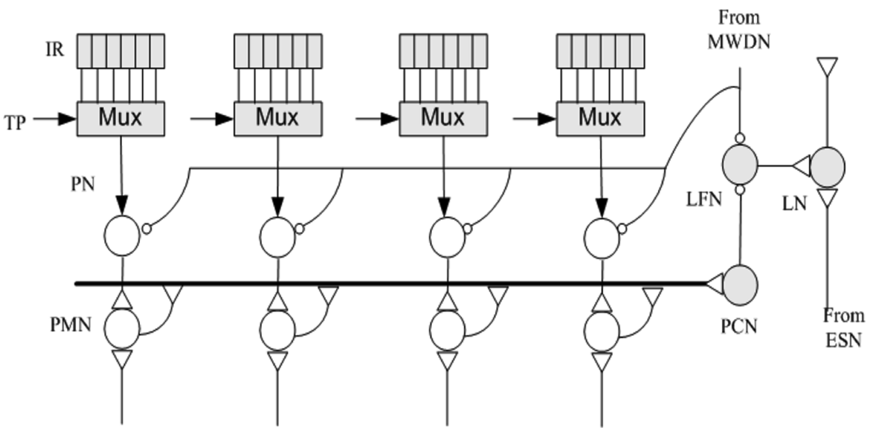


Fig. 9. Prediction mechanism.

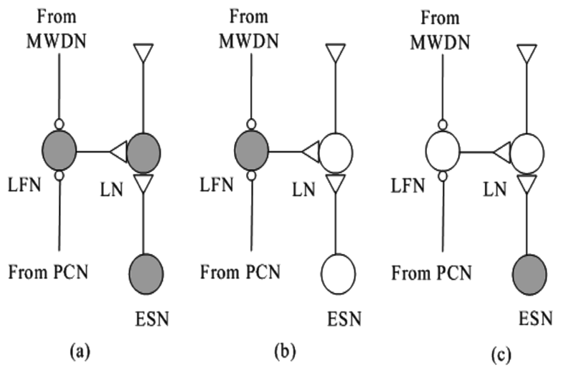


Fig. 10. Firing mechanism of LN.

projections from IR, while neuron n3 receives an inhibitory projection from "i." We now have a single winner with trained links (i.e., Proposition 2). MWDN does not fire, removing the inhibition on PN and LFN. With the control of the TP signal through the MUX, n1 will predict the next symbol as "s," which is not correct in this case. Therefore, PMN will not fire, and PCN does not fire. In this way, LFN will fire because there is no inhibition from PCN. In addition, LN does not fire because ESN does not fire. When the third symbol "t" is presented, both n1 and n3 receive inhibition from IR that stores "t" and neither wins. In this situation, without loss of generality, we may assume that n2 is the winner.

Proposition 4: If there is a single winner in the output neurons with untrained links, a learning signal will be issued at the end of the sequence, and one-shot learning will be executed to learn this sequence.

Proof: In this situation, the MWDN does not fire, and since the winner has untrained links, PN does not fire. Therefore, the PCN does not fire, allowing the LFN to fire. LFN remains active until the ESN fires. The combination of LFN and ESN then causes the LN to fire, and a learning signal activates one-shot learning. Adjusting weights according to the previously described rules forms a representation of "mit" in the memory. Fig. 11 illustrates the strengthened connections after learning.

Propositions 2–4 proved that under all the conditions of the competition stage, the proposed architecture can either correctly predict the sequence, or perform a one-shot learning at the end of a new sequence to learn such sequence. Two matters concerning the prediction mechanism remain to be discussed in Sections II-C2 and II-C3. The first is the mechanism for triggering the activation of the PNs, and the second is the mechanism that implements the time controlled MUX.

2) Activation of a PN: To perform sequence prediction, each IR neuron is associated with a dual IR neuron. WTA neurons responsible for storing the sequence are linked to dual IR neurons through untrained links. IR neurons connect to their dual neurons through trained links. Thus, firing an IR neuron automatically activates its dual neuron. When a sequence is stored in a WTA neuron, connections from the WTA neuron to dual IR neurons corresponding to active IR neurons in the sequence are trained. When a previously stored sequence is input again, a partially matched sequence may activate the WTA neuron of this sequence. This will activate all dual IR neurons that compose the entire sequence. The structure is shown in Fig. 12. This structure combined with the time-controlled MUX provides the mechanism for the prediction scheme.

3) Time-Controlled MUX: The neural network structure of the time-controlled MUX is shown in Fig. 13. The output from WTA activates the dual IR neurons that represent predictions for each time step as discussed in Section II-C2. At a given time step indicated by an active pointer neuron, this dual IR neuron actives the corresponding time-controlled MUX neuron, and, subsequently, the corresponding PN for the next element of the stored sequence. This predicted data is connected to the PMN, which compares the PN with the actual data and fires if the prediction is correct as discussed previously.

Example 4: Assume that through the WTA competition, an output neuron that stores "miss" is firing. Therefore, this neuron will activate its dual IR neurons as discussed in Section II-C2. When the time pointer increases to 3, the upper pointer neuron PT5 is firing. In this case, the concurrently active PT5 neuron and the dual IR neuron (that stores the data "s") will activate the corresponding time-controlled MUX neuron (active neurons are represented by gray circles in Fig. 13). This in turn will activate the corresponding PN, and this activation signal will be send to the PMN to check whether there is a match or mismatch for this predicted data. One should note that Fig. 13 represents the IR

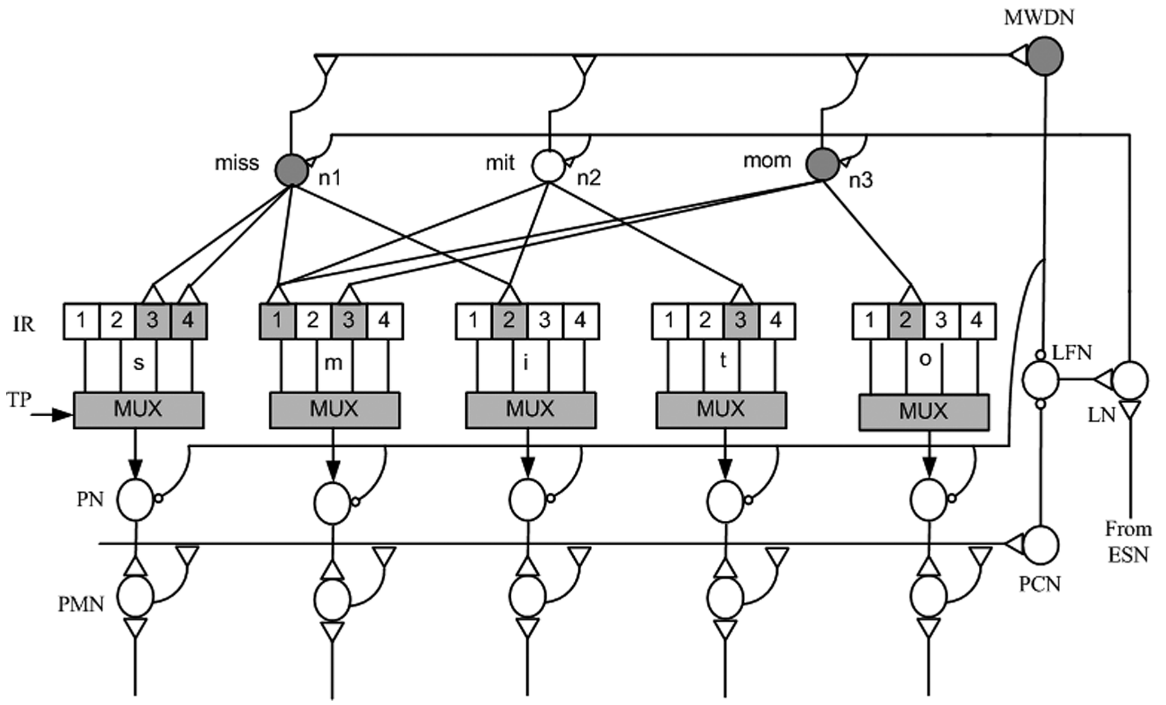


Fig. 11. LTM and multiple winners.

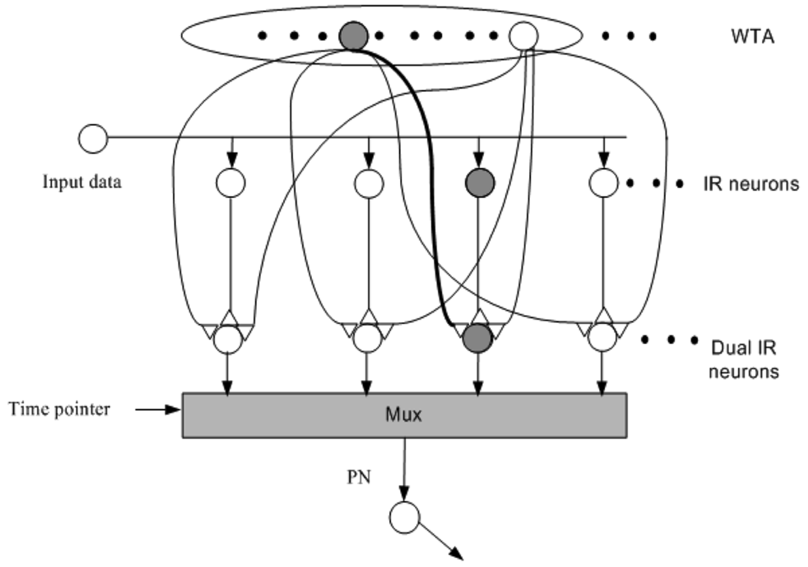


Fig. 12. Activation of PN.

structure that stores letter "s" only, while the IR structures for "m" and "i" are not shown in Fig. 13.

4) Memory Requirements: Let us analyze the memory requirements for this structure. Considering the hierarchical level 1 in Fig. 2, assume that a structure has m_1 input neurons from its lower hierarchical level (i.e., the output neurons form level 0 in Fig. 2). Therefore, the number of PN and PMN will all be m_1. Assume that in this level, the number of output WTA neurons (that provide the input to the next higher hierarchical level) is m_2, and that the longest sequence that this level can store is of the length l. In this case, the total number of IR neurons, dual IR neurons, and MUX neurons will all be equal to $m_1 * l$. The required number of PT neurons will be $2l$ since such PT neurons can be shared by all the IR neurons. Therefore, in this particular hierarchical level, the total required number of neurons will be $2m_1 + m_2 + 3m_1 * l + 2l$.

Let us now estimate the required number of interconnection links. From Fig. 2, we can see that connections between the input of WTA neurons and the dual IR neurons dominate, with the total number of connections equal to $m_1 * m_2 * l$. The connections between the PT neurons inside the IR structure will be $4l - 3$ (see Fig. 6). The other connections include the connections between PMN and the previous hierarchical level output neurons (m_1), PMN and PN(m_1), upper pointer neurons and MUX neurons $(l * m_1)$, and MUX and PN$(l * m_1)$. Therefore, the total required connection will be $(m_1 * m_2 * l) + (4l - 3) + (2 + 2l)m_1$.

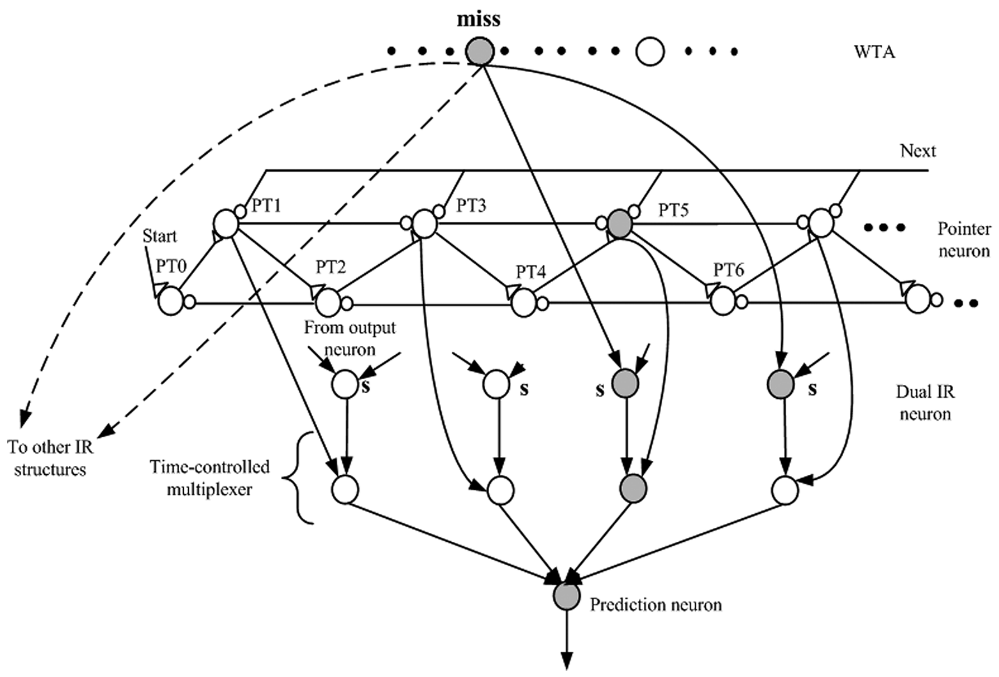


Fig. 13. Time-controlled MUX.

Example 5: Let us assume a hierarchical structure of letter–word–sentence structure. Let us assume that we want to store 10 000 words in the level 2 and 1000 sentences in level 3 (Fig. 2), and the longest word will have ten letters while the longest sentence will have 20 words. Therefore, the total required number of neurons in levels 2 and 3 are $1.08*10^4$ and $6.21*10^5$, respectively, and the required number of interconnections in levels 2 and 3 are $2.6*10^6$ and $2.0*10^8$, respectively.

5) Example of Anticipation-Based Sequence Learning: To this point, we have shown how the model can implement the learning and prediction of a complex sequence. We now give a complete example to summarize how this mechanism works.

Example 6: In order to focus on the sequence learning mechanism, we assume that each sensory input from the environment causes a corresponding winning neuron to fire in the output of the first level. Therefore, we focus on the hierarchical level 1 in Fig. 2. As in [13], let "#" be the end-of-sequence marker for this hierarchical level. Assume that we need to store and retrieve the multiple sequences "mis#mit#miss#mit#," as shown in Fig. 14. Without loss of generality, let this hierarchical level have three output neurons and 27 input neurons (representing the whole alphabet plus the end marker "#"). Each of the output neurons is fully connected to all of the input neurons of all the IRs and their initial weights of the synapses are randomly set to $0.001< w_i <$ 0.01.

When the first symbol of the first sequence "m" is activated, we set the time pointer equal to 1. Since all the weights are initially randomly set, we assume that neuron 1 (N1) is the winner without loss of generality. Since there is no previous training, there is no prediction at this time. Therefore, the PCN does not fire, which causes the LFN to fire. LFN continues to fire until the ESN fires at the end of the first sequence. The combination of LFN and ESN activate the LN, which sets the learning signal. A one-shot learning is triggered and the weights of the winner are adjusted, as discussed in Section II-B (i.e., the excitatory weights are set to 1 and inhibitory weights are set to -100).

When the first symbol of the second sequence "m" is activated, the TP is set to 1. The previously fired neuron (N1) becomes the only winner since it receives all the augmented excitatory projections from the first location of the IR. N1 predicts that the next symbol is "i" through the multiplexer controlled by the TP signal. In this case, the prediction is correct and the corresponding PMN fires, which activates the PCN and inhibits the LFN. When the second symbol "i" is presented to the model, TP is incremented to 2. N1 is still the only winner since it has two excitatory links and no inhibitory links from the first two locations of the corresponding IRs. The corresponding PN predicts "s" as the next input symbol. As this prediction is not true, none of PMN fire, and PCN does not fire. Accordingly, LFN is not inhibited and fires. When the third symbol "t" is activated, TP is increased to 3. N1 is not the winner because it has an inhibitory projection from the IR. Without loss of generality, let us assume that N2 is the winner. When the end of sequence marker "#" is activated, ESN fires. When both LFN and ESN fire, LN fires, which sets the learning signal. A one-shot learning is triggered and the appropriate weights of the winner (N2) are adjusted.

When the first symbol of the third word "m" is presented to the model, there will be two winners with trained connections (N1 and N2). Therefore, MWDN will fire. The firing of MWDN will inhibit all PNs and LFN. MWDN fires, again, when the second symbol "i" is presented. When the third symbol "s" is activated, N1 is the single winner. MWDN does not fire, and the inhibition of PN and LFN is removed. N1 predicts that the next symbol is "#," which is not correct. When the fourth symbol "s" is activated, both N1 and N2 are inhibited. We can assume N3 is the winner without loss of generality. Because N3 has no trained link, there is no prediction. Therefore, PCN does not fire and LFN does fire. The procedure continues until the "#" is

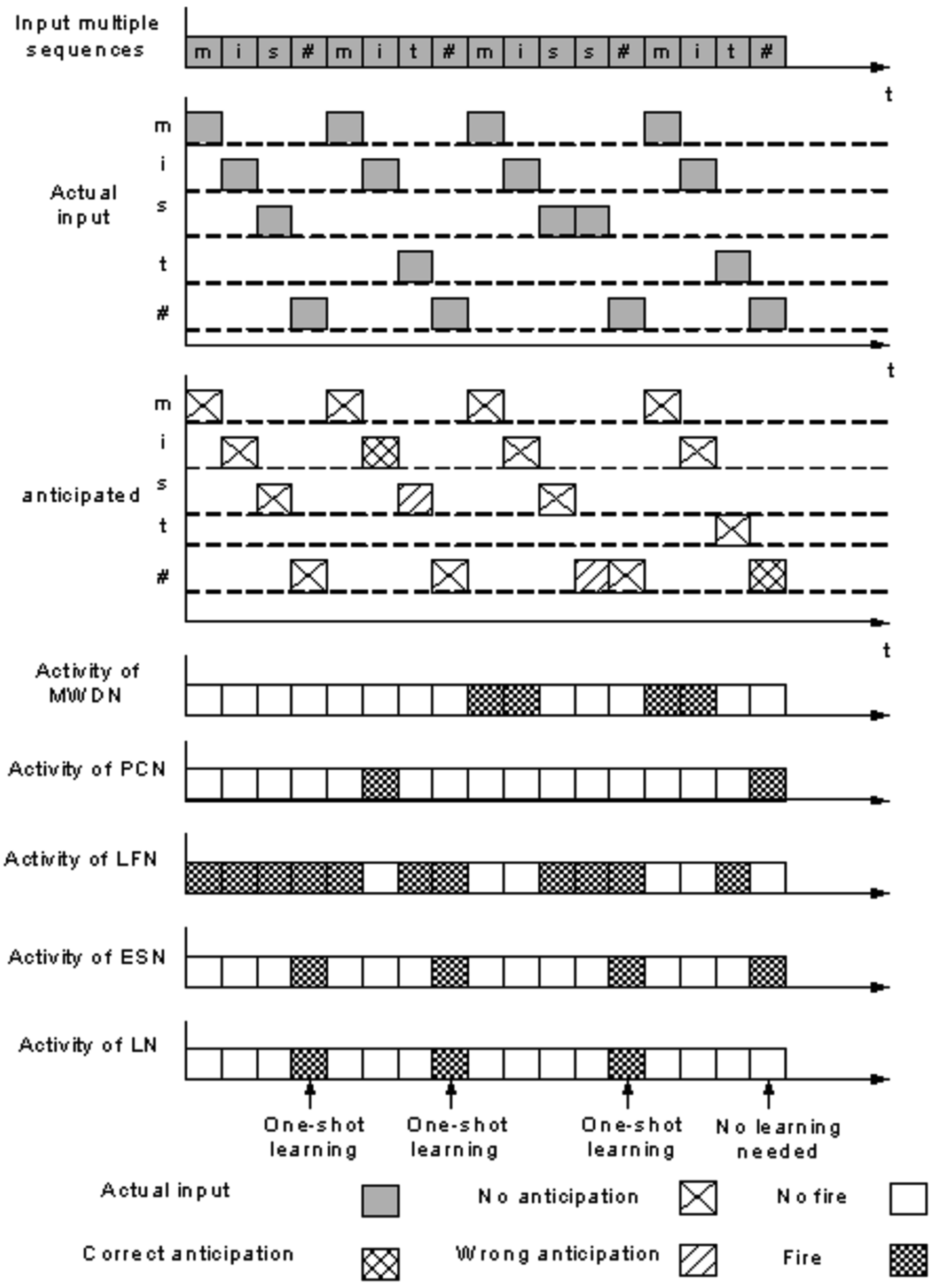


Fig. 14. Learning and anticipation of multiple sequences.

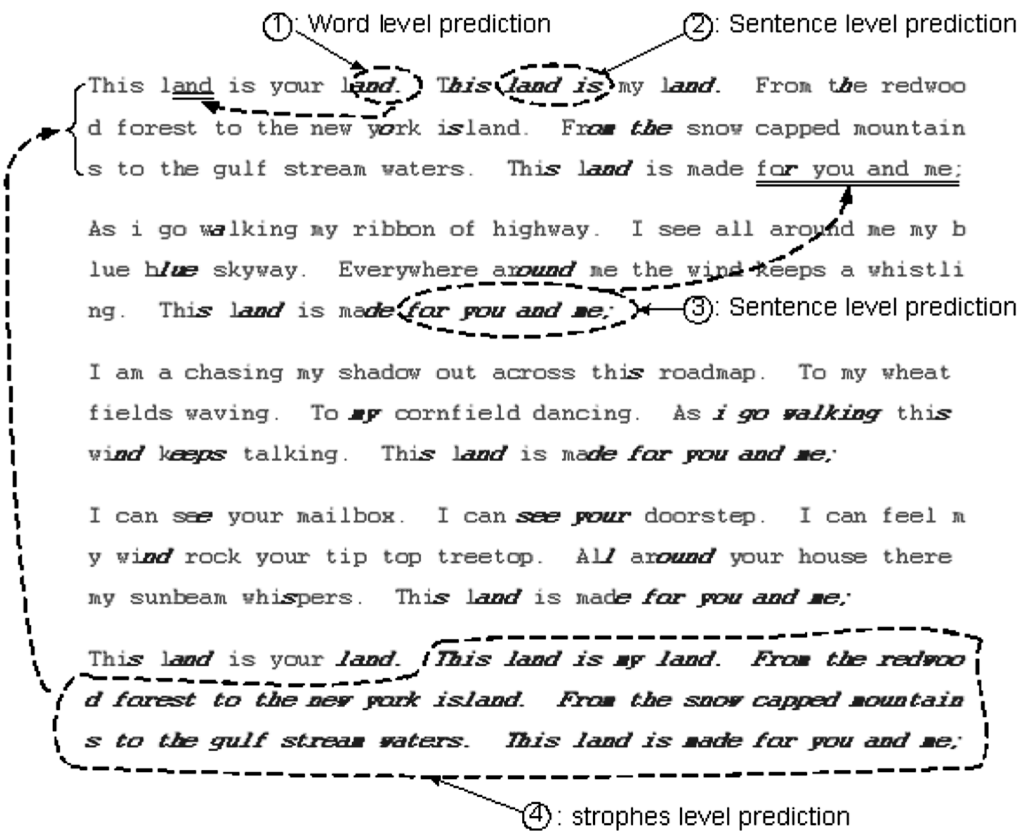
activated and ESN fires. The combined firing of LFN and ESN sends the learning signal, and a one-shot learning is executed, adjusting weights on N3.

When the first symbol of the forth sequence "m" is activated, there are three winners (N1, N2, and N3). MWDN fires and inhibits the PN and the LFN. When the second symbol "i" is activated, these three neurons still are winners. MWDN fires and inhibits PN and LFN. When the third symbol "t" is activated, N2 is the single winner with trained links. MWDN does not fire, and N2 correctly predicts the next symbol "#." When the last symbol of this sequence "#" is activated, ESN fires. Since LFN does not fire when the prediction is correct, LN does not fire. Therefore, no learning is needed for this last sequence.

Fig. 14 shows the firing activities of neurons over the course of the Example 6. The model stored the four sequences in three output neurons (the second and the last sequence being stored in the same neuron: N2). The reader should note that our model does not separate the learning and retrieval processes. We believe that this anticipation-based model is efficient for sequence learning and retrieval.

D. Retrieval of the Stored Sequences

Stored spatio-temporal sequences can be retrieved by activating neurons at any hierarchical level through internal processes or by association with a first level (sensory) input cue. An important element in the process of retrieving the stored


①: Word level prediction
②: Sentence level prediction
③: Sentence level prediction
④: strophes level prediction

This land is your land. This land is my land. From the redwoo
d forest to the new york island. From the snow capped mountain
s to the gulf stream waters. This land is made for you and me;

As i go walking my ribbon of highway. I see all around me my b
lue blue skyway. Everywhere around me the wind keeps a whistli
ng. This land is made for you and me;

I am a chasing my shadow out across this roadmap. To my wheat
fields waving. To my cornfield dancing. As i go walking this
wind keeps talking. This land is made for you and me;

I can see your mailbox. I can see your doorstep. I can feel m
y wind rock your tip top treetop. All around your house there
my sunbeam whispers. This land is made for you and me;

This land is your land. *This land is my land. From the redwoo*
d forest to the new york island. From the snow capped mountain
s to the gulf stream waters. This land is made for you and me;

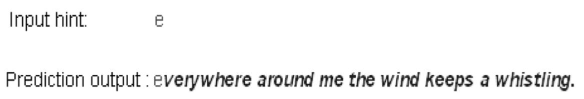
Fig. 15. Simulation results for the temporal sequence learning model.

sequences is the duration of each element of the sequence. By scaling this time, different speeds of presentation may be achieved. Reference [12] describes a mechanism for storing the presentation time intervals in which timing is related to timing of the input sequences. While it is necessary for applications to reproduce the stored sequence at approximately the pace of the original sequence, it is not necessary in general. If a person is given a complex task that requires completion of sequence of operations, timing of the sequence must be determined by the time needed to complete each task, and this timing may be either unknown ahead of time or depend on the context in which the task is executed. Biological organisms commonly rely on sensory-motor coordination to provide proper timing for execution of the stored sequences. The speed of retrieval depends on the natural delay and feedback received through sensory inputs which verifies that a particular element of the sequence was completed. This completion of the sequence element induces presentation of the next element of the stored sequence. Thus, the process of retrieval of stored temporal sequences is self-timed and does not depend on any internal clock, but rather on interaction with the environment. This may be a desired feature for many real life applications where spatio-temporal memories are needed.

III. Simulation Results

A four-level hierarchical structure with letters, words, sentences, and strophes of the popular song "This land is your land" was simulated to illustrate the proposed model. The overall sequence learning model is shown in Fig. 2 with hierarchical levels from level 0 to level N (here $N = 4$). Level 0 was based on the modified Hebbian learning method described in part (1) of Section II-A for pattern recognition. The original sensory input data from the environment contains the scan of $20 * 20$ pixel images for the entire alphabet plus three special characters: space, dot, and semicolon for the end of input sequence for word, sentence, and strophes levels, respectively. There is no distinction between learning and play back, which means that with each incoming sequence, our model will either correctly predict it, or conduct a one-shot learning at the end of the sequence at a particular level. Fig. 15 shows the simulation results for this model. The light gray text is the input sequence and the bold italic text stands for the correctly predicted elements of sequence. ① indicates the correct prediction at the word level. For instance, when the first letter "l" of the second "land" was activated, the memory correctly predicted the next symbol "a," because it had already learned the sequence "land" when presented the first time.

② and ③ in Fig. 15 indicate correct predictions at the sentence level. "This land is made for you and me" is repeated at the end of the second strophe, therefore, the words "for you and me" were correctly predicted when "This land is made" was presented. The reason "land is made for you and me" cannot be predicted after "This" is that we have some sentences, such as "This land is your land" "This land is my land" with the same first word. Accordingly, the MWDN neuron fired after "This" was presented, inhibiting firing of the PN. ④ in Fig. 15 indicates a correct prediction at the strophes level. The last strophe



```
Input hint:            e

Prediction output : everywhere around me the wind keeps a whistling.
```

Fig. 16. Prediction result based on hint.

is the repetition of the first one. Therefore, after the first sentence of the strophe, the model correctly predicted the rest of the strophe.

After the model has stored this song, we can also check its ability to predict the song from an input hint, or to learn a new sequence without destroying the stored sequence. For example, if we give the hint "e" to the system, there is a unique neuron in the word level that stores the word "everywhere; " so, the model will play the word "everywhere." After this neuron in the word level fires, it will trigger a neuron in the sentence level to fire. In this case, a unique neuron in the sentence level will be the winner, and that neuron will play back its stored sequence from the sentence level, to the word level, and finally generate the original sequence. Fig. 16 shows the simulation result after the character "e" was activated as a hint to the model.

In addition, if a new word, sentence or strophe is presented to our model, the model can further learn this new sequence without destroying the previously stored sequence, and this learned sequence can be used for associative predictions.

Since many intelligent behaviors are in the sequential format, including reasoning, planning, scheduling, and goal driven learning, the presented hierarchical sequence learning model has wide application potentials. For instance, this model can be used for intelligent word processing, or sequential audio and video signal processing. The model can learn a person's speaking style and later can recognize and predict his speech based on some clues. The most important applications of temporal sequence learning are in the machine intelligence models. Such models require constant predictions of incoming information based on the observed in the past sequences of events. Temporal sequence learning is needed to perform complex sequences of movements or action in navigating robots or completing complex tasks in interaction with the environment.

IV. Conclusion

In this paper, a novel temporal sequence learning model was presented. This model is characterized by hierarchical organization and anticipation-based learning. A modified Hebbian learning mechanism was proposed for input pattern recognition at the lowest level of the model. In each hierarchical level, a WTA mechanism was used to select neurons that serve as the input for the next higher hierarchical level. The resulting model can efficiently handle large scale, multiple, complex sequence learning, and retrieval. Prediction is an essential component of the temporal sequence learning in our model. By correct prediction, the machine indicates it knows the current sequence and does not require additional learning. Otherwise, one-shot learning is executed. To show the effectiveness of the proposed model, a four-level hierarchical structure of letters, words, sentences, and strophes was simulated for this paper.

The results reported in this paper suggest that this temporal sequence learning model can be an essential element of an intelligent machine, and may be of interest for the construction of engineered devices. As indicated in the Introduction, we are also interested in biological memory and intelligence. The use, in this model, of WTA and single neuron representations of symbols (so called "grandmother cells") flies in the face of the widely held view that real brains use distributed representations, perhaps consisting of a few hundred neurons. The use of grandmother cells prevents the model from showing the "graceful degradation" of memory seen with loss of neurons from real brains. The model also seems too highly engineered to be an evolved device, and there is little in the real environment to serve as start- or end-of-sequence signals. However, this model is only a simplification of the biologically plausible process in which the grandmother cells are replaced by groups of neurons firing synchronously, as it is suggested in [30]. Groups of neurons organized in minicolumns are able to perform feature selection and classification in structures that resemble WTA operation of grandmother cells as demonstrated in [31]. In addition, sufficient computational resources should make it possible to render the present model in large associative networks that use sparsely coded, distributive representations. For example, as in the model presented here, R-nets [32], [33] implement a mechanism of associative recall that depends on the absence of inhibitory projections. The biological plausibility of the mechanism is argued in [33]. Nearly identical R-nets have already been assembled in a modular fashion to create the sort of "and," "or," and "if not" gates needed in the present model. As the function of an R-net depends chiefly on its position in the network, the use of such networks would reduce the appearance of *ad hoc* engineering. Finally, by making the persistence of synaptic training in an assembly of R-nets dependent on the frequency of repetition of training events, it is possible to for the network to learn a sequence without including the various events that preceded or followed the sequence on different occasions [34]. Accordingly, the requirement for start- and end-of-sequence codes is relieved.

References

[1] R. Sun and C. L. Giles, "Sequence learning: From recognition and prediction to sequential decision making," *IEEE Intell. Syst.*, vol. 16, no. 4, pp. 67–70, Jul./Aug. 2001.

[2] J. H. Wang, M. C. Tsai, and W. S. Su, "Learning temporal sequences using dual-weight neurons," *J. Chinese Inst. Eng.*, vol. 24, pp. 329–344, 2001.

[3] N. Y. Liang, G. B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[4] A. G. Tijsseling, "Sequential information processing using time-delay connections in ontogenic CALM networks," *IEEE Trans. Neural Netw.*, vol. 16, no. 1, pp. 145–159, Jan. 2005.

[5] J. L. Elman, "Finding structure in time," *Cognitive Sci.*, vol. 14, pp. 179–211, 1990.

[6] M. I. Jordan, "Attractor dynamics and parallelism in a connectionist sequential machine," in *Proc. 8th Annu. Conf. Cognitive Sci. Soc.*, 1986, pp. 531–546.

[7] J. B. Pollack, "The induction of dynamical recognizers," *Mach. Learn.*, vol. 7, pp. 227–252, 1991.

[8] J. Tani, "Learning to generate articulated behavior through the bottom-up and the top-down interaction process," *Neural Netw.*, vol. 16, pp. 11–23, 2003.

[9] J. Tani and S. Nolfi, "Learning to perceive the world as articulated: An approach for hierarchical learning in sensory-motor systems," *Neural Netw.*, vol. 12, pp. 1131–1141, 1999.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, 1997.

[11] D. Wang and M. A. Arbib, "Complex temporal sequence learning based on short-term memory," *Proc. IEEE*, vol. 78, no. 9, pp. 1536–1543, Sep. 1990.

[12] D. Wang and M. A. Arbib, "Timing and chunking in processing temporal order," *IEEE Trans. Syst., Man, Cybern.*, vol. 23, no. 4, pp. 993–1009, July/Aug. 1993.

[13] D. Wang and B. Yuwono, "Anticipation-based temporal pattern generation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 4, pp. 615–628, Apr. 1995.

[14] D. Wang and B. Yuwono, "Incremental learning of complex temporal patterns," *IEEE Trans. Neural Netw.*, vol. 7, no. 6, pp. 1465–1481, Nov. 1996.

[15] L. Wang, "Learning and retrieving spatio-temporal sequences with any static associative neural network," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 45, no. 6, pp. 729–739, Jun. 1998.

[16] L. Wang, "Multi-associative neural networks and their applications to learning and retrieving complex spatio-temporal sequences," *IEEE Trans. Syst., Man, Cybern B, Cybern.*, vol. 29, no. 1, pp. 73–82, Feb. 1999.

[17] S. Chartier and M. Boukadoum, "A sequential dynamic heteroassociative memory for multistep pattern recognition and one-to-many association," *IEEE Trans. Neural Netw.*, vol. 17, no. 1, pp. 59–68, Jan. 2006.

[18] J. Bose, S. B. Furber, and J. L. Shapiro, "An associative memory for the on-line recognition and prediction of temporal sequences," in *Proc. Int. Joint Conf. Neural Netw.*, Montreal, QC, Canada, Jul. 31–Aug. 4 2005, pp. 1223–1228.

[19] C. G. N. Manning and I. H. Witten, "Identifying hierarchical structure in sequences: A linear-time algorithm," *J. Artif. Intell. Res.*, vol. 7, pp. 67–82, 1997.

[20] D. George and J. Hawkins, "Invariant pattern recognition using Bayesian inference on hierarchical sequences" [Online]. Available: http://www.stanford.edu/~dil/RNI/DilJeffTechReport.pdf

[21] S. Grossberg, "Competitive learning: From interactive activation to adaptive resonance," *Cognitive Sci.*, vol. 11, pp. 23–63, 1987.

[22] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Phys. Learn. Motivation*, vol. 24, pp. 109–165, 1989.

[23] N. E. Sharkey and A. J. C. Sharkey, "Understanding catastrophic interference in neural nets," Dept. Comput. Sci., Univ. Sheffield, U.K., Tech. Rep. CS-94-4, 1994.

[24] C. A. Kortge, "Episodic memory in connectionist networks," in *Proc. 12th Annu. Conf. Cognitive Sci. Soc.*, 1990, pp. 764–771.

[25] R. M. French, "Using semidistributed representations to overcome catastrophic forgetting in connectionist networks," in *Proc. 13th Annu. Conf. Cognitive Sci. Soc.*, 1991, pp. 173–178.

[26] S. A. Sloman and D. E. Rumelhart, "Reducing interference in distributed memories through episodic gating," in *From Learning Theory to Connectionist Theory: Essays in Honor of William K. Estes*, A. F. Healy, S. M. Kosslyn, and R. M. Shiffrin, Eds. Hillsdale, NJ: Lawrence Erlbaum, 1992, pp. 227–248.

[27] G. H. Bower, S. Thompson-Schill, and E. Tulving, "Reducing retroactive interference: An interference analysis," *J. Exp. Psych: Learning, Memory, Cogn.*, vol. 20, pp. 51–66, 1994.

[28] K. Doya and S. Yoshizawa, "Memorizing hierarchical temporal patterns in analog neuron networks," in *Proc. Int. Joint. Conf. Neural Netw.*, San Diego, CA, 1990, vol. 3, pp. 299–304.

[29] P. J. Jennings and S. W. Keele, "A computational model of attentional requirements in sequence learning," in *Proc. 12th Ann. Conf. Cog. Sci. Soc.*, 1990, pp. 876–883.

[30] D. Wang, J. Buhman, and C. von der Malsburg, "Pattern segmentation in associative memory," *Neural Comput.*, vol. 2, pp. 94–106, 1990.

[31] J. Lücke and C. von der Malsburg, "Rapid processing and unsupervised learning in a model of the cortical macrocolumn," *Neural Comput.*, vol. 16, pp. 501–33, 2004.

[32] D. Vogel, "A biologically plausible model of associative memory which uses disinhibition rather than long term potentiation," *Brain Cogn.*, vol. 45, pp. 212–228, 2001.

[33] D. Vogel, "A neural network model of memory and higher cognitive functions," *Int. J. Psychophysiol.*, vol. 55, pp. 3–21, 2005.

[34] D. Vogel, Personal Communication.

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

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

Haibo He (M'06) received the B.S. and M.S. degrees in electrical engineering from Huazhong University of Science and Technology (HUST), Wuhan, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from Ohio University, Athens, in 2006.

He is an Assistant Professor at the Department of Electrical and Computer Engineering at Stevens Institute of Technology, Hoboken, NJ. His research interests include intelligent systems design, very large scale integration (VLSI) and field-programmable gate array (FPGA) design, bioinspired computing architecture, and computational intelligence and their applications.