Sequence Recognition with Spatio-Temporal Long-Term Memory Organization

Vu-Anh Nguyen, *Student Member, IEEE*, Janusz A. Starzyk, *Senior Member, IEEE*, and Wooi-Boon Goh, *Member, IEEE*

***Abstract*—In this work, we propose a connectionist memory structure for spatio-temporal sequence learning and recognition inspired by the Long-Term Memory structure of human cortex. Besides symbolic data, our framework is able to continuously process real-valued multi-dimensional data stream. This capability is made possible by addressing three critical problems in spatio-temporal learning, namely error tolerance, significance of sequence's elements and memory forgetting mechanism. We demonstrate the potential of the framework with a synthetic example and a real world example, namely the task of hand-sign language interpretation with the Australian Sign Language dataset.**

***Keywords*—Hierarchical memory architecture, spatio-temporal neural networks, hand-sign language interpretation.**

I. Introduction

Modeling of sequential memory is vital for developing many aspects of embodied intelligent system [1]. In this work, we introduce a spatio-temporal memory architecture which has an efficient storage mechanism and is able to learn sequential data in a flexible manner. We propose a novel recognition algorithm that can robustly recognize test sequences with various degrees of distortions. Our architecture is also capable of working with continuous multi-dimensional data stream instead of just symbolic type of data as in [2, 3]. The capability and performance of the framework for sequence recognition are demonstrated with a synthetic working example and the Australian Sign Language (ASL) dataset [4].

The general concepts in sequence learning including prediction, recognition and decision making were reviewed by Sun and Giles in [5]. From a connectionist perspective, Kremer [6] presented a comprehensive classification of neural network approaches in learning spatio-temporal patterns. The models of spatio-temporal neural architecture mainly involve two types of memories, namely Short-Term Memory (STM) and Long-Term Memory (LTM). STM is used as a temporal storage of input data for rapid processing and has a limited capacity [7]. Additionally, STM stores the order of input elements and is subjected to interference and decay over time. On the other hand, LTM is built based on synaptic modifications of neural activities of STM [8]. The content and temporal order of a sequence in an LTM structure are stored explicitly as synaptic weights [9]. In this work, we follow a similar design principle which is based on interaction of the two types of memories.

Research on spatio-temporal neural network dates back to the out-star avalanche model and STORE model [10]. Lang and Hinton [11] introduced the time delay neural network (TDNN). TDNN learns a sequence as a static feed-forward network using the back-propagation through time (BPTT) algorithm as the training method and replicated neurons as time delays. Later, recurrent neural networks (RNNs) were proposed to address the drawback of the explicit time representation by TDNN. RNNs employ internal feedback links and a temporary buffer of recent states. Two popular RNN models are [5] and [6]. Subsequent improvement were presented in [12] with Long-Short Term Memory (LSTM).

For static neural network architectures, Wang et al [2, 13, 14] introduced several key properties of complex sequence analysis including temporal chunking, storage, recognition, hierarchical organization and incremental learning. Our previous model [15] developed a sequence learning model which concentrates on several critical properties of LTM such as competition, hierarchical organization, anticipation and one-shot learning. Similarly to [14], for incremental learning, the network actively predicts next element. Learning of a new sequence occurs when there is a sufficient mismatch with the stored sequence. However, the main difference is that the sequence chunking process is done automatically. In addition, the training stage requires only a single pass of training sequence because of the adoption of a one-shot learning strategy. Evaluation with storage and generation/prediction of text paragraphs demonstrate the effectiveness of the proposed system.

In this work, we extend our previous work in [3, 15] and focus on a number of crucial aspects that must be addressed to achieve robust sequence recognition for processing real-valued and multidimensional sequences. The first factor is the error tolerance within spatio-temporal patterns. The second factor is the incorporation of significance of elements in the LTM cell. The third is the augmentation of the framework with a novel activation decay mechanism.

Errors in sequence analysis can be broadly categorized into two types: Inter-element and intra-element. The former includes various distortions of temporal relationship among consecutive elements. The latter refers to various distortions in the content of the input. For intra-element error, we characterize the error tolerance of each element by estimating the statistical spatio-temporal variation. For inter-element error, the error tolerance of consecutive elements is characterized by a sequence matching algorithm which is capable of handling inter-element variability. When a test

Vu-Anh Nguyen and Wooi-Boon Goh are with the School of Computer Engineering, Nanyang Technological University, Singapore.

. Janusz A. Starzyk is with the School of Electrical Engineering and Computer Science, Ohio University, Athens, USA.

978-1-4673-1490-9/12/$31.00 ©2012 IEEE

sequence is presented, LTM cells incrementally accumulate evidences from the testing sequence and compete to be the winner. Only the stored sequence in an LTM cell elicits a maximum activation. Putting it differently, any deviation from the ideal sequence results in a corresponding degradation in activation.

The significance of elements stored in each LTM cell is an important issue in spatio-temporal learning. Due to the limited computational resource, an agent may choose to put more emphasis on identifying and processing only an important subset of sequence elements. This complements the one-shot learning that assigns unit significance to all the elements. The novelty of our model is the explicit modulation of the LTM activation by the estimated elements' significance. We propose a specific significance analysis that is suitable for the chosen applications based on the statistical variation of the elements. It is well understood that the definitions and identification techniques of significant elements vary depending on specific applications. In this work, the significance to the LTM activation is integrated as a modulatory factor for any chosen significance's estimations method.

The last contribution of our network structure is the introduction of memory activation decay. The reasons for the activation decay are two-fold. The first is to maintain the strength of an LTM cell for a sufficient duration to perform learning, construct associations and predict subsequent events. The second is that when the current sequence of events increasingly deviates from the LTM cell, the LTM cell's output strength needs to decay rapidly to avoid ambiguities in decision making. The memory forgetting problem has been discussed previously in neuro-psychological, neuro-physiological [16] and their computational aspects [3, 13].

The structure of the paper is as follows. Section II summarizes the main characteristic of the hierarchical Long-Term Memory architecture and presents the neural architecture and recognition algorithm of the proposed LTM cell structure. Section III presents empirical studies using the synthetic data and the ASL dataset. In the case of ASL dataset, we present a series of classification experiments based on a nearest-neighbor classifier and a study on sequences alignment and combination using the proposed LTM model. Finally, section IV concludes the paper.

II. LONG-TERM MEMORY ORGANIZATION

A. Overview of hierarchical LTM architecture

The hierarchical LTM memory architecture follows a number of critical properties of human cortex's structure including hierarchical, uniform, sparse and redundant representation [17]. Hierarchical representation provides a flexible way to tackle large and complex problems in which the high-level layers exploit the low-level ones to learn a large number of different patterns [18].

The hierarchy contains multiple layers of LTM cells. The first layer, STM, serves as a temporary buffer between an input vector and the first LTM layer. The input to the network is represented as a multi-dimensional feature vector extracted from the environment via multi-modal sensory systems. Each LTM layer contains a number of identical complex cells, i.e. LTM cells, with possibly overlapping output regions. In each output region, multiple winners are selected through competition and mutual inhibition by k-WTA rule. This yields a sparse representation at each level of hierarchy. In each LTM cell, the content of STM updates the activation of the LTM cell accordingly. Each LTM layer also contains several feedback links from higher layers for deployment of anticipation, pattern completion and attention. The outputs of LTM cells at each layer are fetched to the next higher layer for more complex processing. This process results in chunking of a long sequence into short segments activated on a lower level. LTM activations from multiple cells can also be combined for ensemble decision making strategies at each layer.

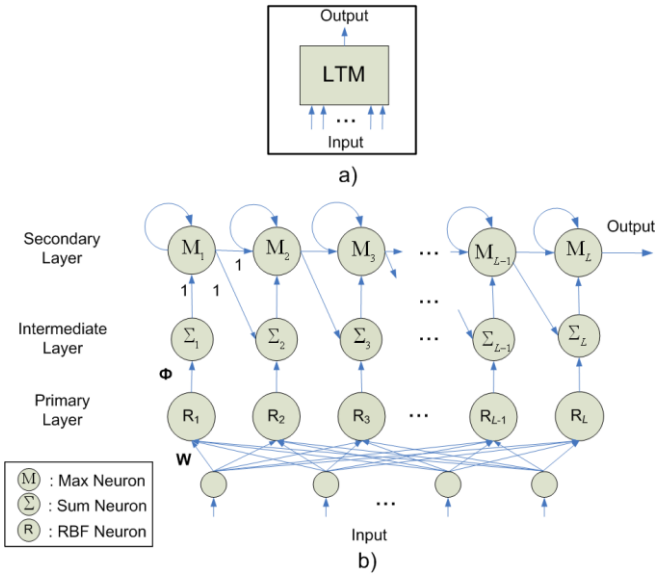

Figure 1: LTM Structure

B. Mathematical Notations

In this paper, a spatio-temporal sequence **S** is represented as $S_1 - S_2 - \cdots - S_L$, where S_i is an element of the sequence and L is the length of the sequence. Each element of the sequence **S** is represented by a multi-dimensional vector, i.e. $S_i \in \mathbb{R}^F$ where F is the dimension. In a matrix form, the sequence **S** can be represented as

$$\mathbf{S} \equiv \{S_{ij} | \, i = 1 \ldots F, j = 1 \ldots T\} \quad (1)$$

A subsequence of **S** is any $S_m - S_{m+1} - \cdots - S_n$, where $1 \le m \le n \le L$. We denote the tolerance $\boldsymbol{\Delta} \in \mathbb{R}^{F \times T}$ in a matrix form and the significance $\boldsymbol{\phi} \in \mathbb{R}^L$ in a vector form of the elements of a sequence **S** as follows:

$$\boldsymbol{\Delta} \equiv \{\delta_{ij} \in \mathbb{R}^+ | i = 1 \ldots F, j = 1 \ldots L\} \quad (2)$$

$$\boldsymbol{\Phi} \equiv \{\phi_{ij} \in (0,1] | \, j = 1 \ldots L\} \quad (3)$$

The tolerance and significance of a sequence describe the statistical variation and the importance of the elements in a sequence respectively. The previous work in [3] only dealt with the case when $\delta_{ij} \approx 0 \; \forall i, j$ (i.e. either match or non-

match) and $\phi_j = c\ \forall j$ where c is a constant (i.e. uniform significance).

C. LTM Structural Organization

In this section, we describe the structural organization of an LTM cell and its storage and recognition mechanisms of spatio-temporal sequences. The block diagram of an LTM cell is depicted in Fig. 1a. Each LTM cell stores a sequence $\mathbf{S} \in \mathbb{R}^{F\times T}$ as synaptic weight $\mathbf{W} \in \mathbb{R}^{F\times T}$. The network structure comprises of 4 layers:

- Input Layer: The input layer consists of F input neurons which correspond to an F-dimensional input vector at a time step t:

$$\mathbf{I}(t) \equiv \{I_i(t) | i = 1 \dots F\} \tag{4}$$

The input vector can either come from sensory systems connected to the environment, outputs of lower LTM layers or feedbacks from higher LTM layers.

- Primary Layer: The primary layer consists of L primary neurons (PNs) ("R" neurons in the Fig. 1b). In this structure, the content of a sequence is stored as the synaptic weight matrix $\mathbf{W}$ connecting the input layer and the primary layer:

$$\mathbf{W} \equiv \{w_{ij} | i = 1 \dots F, j = 1 \dots L\} \tag{5}$$

The primary layer computes the similarity between an input vector $\mathbf{I}(t)$ and each element of the sequence. In this work, the radial basis function is employed as the similarity metric as follows:

$$y_j^{PN}(t) = \exp\left\{-\frac{1}{F}\sum_{i=1}^{F}\frac{\left(w_{ij}-I_i(t)\right)^2}{\delta_{ij}^2}\right\} \tag{6}$$

where $y_j^{PN}(t)$ is the output (or *primary excitation*) of the j^{th} PN neuron at time t induced by the input $\mathbf{I}(t)$ and δ_{ij} is the tolerance of the i^{th} feature of the j^{th} element.

- Intermediate Layer: The intermediate layer consists of L intermediate neuron (INs) ("$\sum$" neuron in the Fig. 1b). At a time step t, the activation of j^{th} IN, denoted as $y_j^H(t)$, combines the outputs of the j^{th} PN and the $(j-1)^{th}$ SN (described later) computed at the previous step (denoted by $y_{j-1}^{SN}(t-1)$). The connection between each PN and IN is weighted by the significance of the corresponding element:

$$y_j^H(t) = \left[\phi_j y_j^{PN}(t) + \hat{y}_{j-1}^{SN}(t)\right]^+ \tag{7}$$

where $\hat{y}_j^{SN}(t)$ is the activation of the j^{th} SN after being decayed and $[x]^+ = x$ if $x \geq 0$ and 0 otherwise. The neuronal decaying behavior of the SNs is modeled by the function $f_j(x)(j = 1 \dots L)$, depicted as the self-feedback loop in the Fig. 1b:

$$\hat{y}_j^{SN}(t) = f_j\left(y_j^{SN}(t-1)\right) \tag{8}$$

The decaying function satisfies the condition: $f_j(x) \leq x, \forall j = 1 \dots L$.

- Secondary Layer: The secondary layer consists of L secondary neurons (SNs) ("M" neurons in the Fig. 1b). The activation of SNs is updated incrementally as follows:

$$y_j^{SN}(t) = \max\{\hat{y}_j^{SN}(t), y_j^H(t), y_{m-1}^{SN}(t)\} \tag{9}$$

where $y_j^{SN}(t)$ is the activation of of the j^{th} SN at the time step t (or *secondary excitation*) and max{.} is the point-wise maximum. The updated activation of the m^{th} SN provides a matching degree between a test sequence and the subsequence $S_1 - S_2 - \dots - S_j$ of the stored sequence. The activation of the j^{th} SN is computed based on the maximum contribution from three different signals: decayed activation from the j^{th} from the previous step $\hat{y}_j^{SN}(t)$, newly updated j^{th} IN$y_j^H(t)$ and the current matching degree between the presented sequence and the subsequence $S_1 - S_2 - \dots - S_{j-1}$ of the stored sequence.

In this work, we use a linear decaying function for modeling the decaying behavior, which is given by:

$$f_j(x) = x - \gamma\phi_j \tag{10}$$

where γ is the decaying rate ($\gamma \in [0,1]$). Non-linear decay typically requires specialized knowledge of the learning sequence which is not assumed in this paper.

The output of an LTM cell at a time step t is given by the secondary excitation of the last SN, i.e. $y_L^{DN}(t)$. This activation provides a matching degree between an input sequence presented until the time step t and the sequence stored in the LTM cell. The maximum activation of an LTM cell is attained by presenting the LTM cell with the stored sequence. The analytical expression for the maximum activation of an LTM cell is given by:

$$O_{max} = \max\{y_L^{SN}(t)\} = (1-\gamma)\sum_{j=1}^{L}\phi_j \tag{11}$$

This maximum activation is used to normalize the LTM activation between 0 and 1. Therefore, it allows the comparison of a test sequence that is of a different length to the stored sequence.

D. LTM Storage and Learning

Using one-shot learning, an LTM cell learns a sequence $\mathbf{S} \in \mathbb{R}^{F\times T}$ as synaptic connection weight $\mathbf{W} \in \mathbb{R}^{F\times T}$. The learning of a new sequence occurs only when no LTM cell elicits a sufficient matching degree to the test sequence, i.e. the maximum LTM activation is below a threshold κ. In an intelligent system, the threshold κ of an LTM cell is determined via interaction with the environment. In this paper, we set the threshold to 0 unless otherwise stated.

One shot learning has been shown to improve training efficiency since it requires only a single presentation of a target sequence. It is also critical for some important applications of LTM cells, for instance the organization of episodic memory [19]. The gradual Hebbian-like synaptic modification instead of the one-shot storage was discussed in [13]. Hierarchical and distributed representation with chunking can be incorporated to improve the storage

capacity. However, in the following only a single layer of LTM is considered for simplicity.

E. LTM Recognition

This section develops a sequence recognition algorithm called LTM Sequence Recognition (LTMSR) (Algorithm 1, Fig. 2) based on the architecture. Each input vector of a test sequence is incrementally presented to an LTM cell. Once the matching output is returned, a winning LTM sequence can be determined by a WTA network of the existing LTM cells.

The LTMSR introduces the delay factor τ and corresponding counters, $C \equiv \{c_j | \, j = 1 \dots L\}$, which retain the SNs' activations for a number of steps before being reset. In this work, the maximum delay τ is set to 1 for all experiments. The purpose of the delay factor is to compensate for minor delay or perturbation of input $\mathbf{I}(t)$. The computational complexity of the algorithm is in the order of $O(L)$ where L is the length of the LTM cell.

[Algorithm 1: LTM Sequence Recognition]

Require: $\mathbf{W}, \boldsymbol{\phi}, \boldsymbol{\Delta}, O_{max}, \gamma$

Ensure: O

Initialize:

- $y_j^{SN}(0) \leftarrow 0 \forall j = 1 \dots L$
- $c_j \leftarrow \tau \; \forall j = 1 \dots L$
- $t \leftarrow 0$

Start Algorithm:

For each input vector $I(t)$ of a test sequence do:
  Compute $y_j^{PN}(t), \hat{y}_j^{SN}(t)$ and $y_j^H(t) \; \forall j = 1 \dots L$.
  For m = 1 to L do
    If $(\hat{y}_j^{SN}(t) \geq \max\{y_j^H(t), y_{j-1}^{SN}(t)\}$ and $c_j \geq 0$ then
      $y_j^{SN}(t) \leftarrow \hat{y}_j(t)$
      $c_j \leftarrow c_j - 1$
    Else
      $y_j^{SN}(t) \leftarrow \max\{y_j^H(t), y_{m-1}^{SN}(t)\}$
      $c_j \leftarrow \tau$
    End If
  End For
  $t \leftarrow t + 1$
End For
Return $O = \frac{y_L^{SN}(t)}{O_{max}}$

[End Algorithm]

Figure 2: LTMSR Algorithm

F. Intra-element tolerance characterization

In this section, we propose an adaptive characterization of spatial uncertainties based on the local variations of features. The estimated uncertainty is used to normalize the matching between each LTM element and an input vector as in (6). Given the synaptic connection $\mathbf{W}$ of an LTM cell, the local standard deviation (LSD) of elements with respect to the time dimension is employed to estimate $\boldsymbol{\Delta}$. The LSD is estimated over a local window $\boldsymbol{\Psi}_j$ $(j = 1 \dots L)$ of size $(2\psi + 1)$ where ψ is an integer.

$$\delta_{ij} = \sqrt{\frac{1}{2\psi+1} \sum_{k \in \boldsymbol{\Psi}_j} \left(w_{ik} - \mu_{ij}\right)^2} \tag{12}$$

where μ_{ik} is the mean of the i^{th} feature with respect to the local window $\boldsymbol{\Psi}_j$.

$$\mu_{ik} = \frac{1}{2\psi+1} \sum_{k \in \boldsymbol{\Psi}_j} w_{ik} \tag{13}$$

In this case we assume that the features are independent. The influence of covariance of features towards robust tolerance estimation is currently under investigation. In the following, the window parameter ψ is set to 5 unless otherwise stated.

G. Significance of sequence's elements

Significance estimation provides an evaluation of the importance of each element within an LTM cell, which helps the LTM cell to focus on identifying highly distinguishing elements of a sequence. The significance of elements $\boldsymbol{\Phi}$ is integrated to modulate the activation of an LTM cell.

The proposed significance estimation in this work proceeds from the feature level to the element level. Given an LTM cell, we denote the mean and standard deviation of the i^{th} $(i = 1 \dots L)$ feature as μ_i and σ_i respectively.

$$\mu_i = \frac{1}{L} \sum_{k=1}^{L} w_{ik} \tag{14}$$

$$\sigma_i = \sqrt{\frac{1}{L-1} \sum_{k=1}^{L} (w_{ik} - \mu_i)^2} \tag{15}$$

The significance estimation of an LTM cell at the feature level is denoted as:

$$\mathbf{R} \equiv \{r_{ij} | i = 1 \dots F, j = 1 \dots L\} \tag{16}$$

and is computed as:

$$r_{ij} = 1 - \exp\left\{-\frac{(w_{ij} - \mu_i)^2}{\eta \sigma_i^2}\right\} \tag{17}$$

where η is a tuning parameter. Finally, the significance of the LTM cell at the element level ($\boldsymbol{\Phi}$) is computed as:

$$\phi_j = \sqrt{\frac{\sum_{k=1}^{F} r_{kj}^2}{F}} \tag{18}$$

From (18), we have $r_{ij} \in [0,1] \forall i, j$, therefore $\phi_j \in [0,1] \forall j$.

Intuitively, the significance as in (18) gives high values to the elements which have the feature values statistically different from the mean values and vice versa. As mentioned earlier, the proposed significance estimation is suitable for our chosen application but may need to be re-formulated for other domains with different data characteristics.

III. EXPERIMENTS

A. Experiment 1: A synthetic example

We consider the 2D sequence $\mathbf{S} \equiv ABCD$ with the length of 4. Each element of the sequence is specified as follows: $A = (0.2\,;\, 0.8)$, $B = (0.4\,;\, 0.6)$, $C = (0.6\,;\, 0.4)$, $D = (0.8\,;\, 0.2)$. The sequence S is stored as an LTM cell by one-shot learning. Therefore, there are 2 neurons in the Input layer and 4 neurons in each of the Primary, Intermediate and Secondary layer. The specifications of the LTM cell are set as follows: $\delta_{ij} = 0.1 \forall i = [1,2]$, $j \in [1,4]$, $\phi_j = 1 \forall j \in [1,4]$.

A number of test sequences are synthesized based on the stored sequence to evaluate the robustness of the LTM cell's activation. The result is shown in Table 1. The original sequence (Type 0) and four types of sequential distortions including order distortion (Type 1), replicated elements (Type 2), missing elements (Type 3) and noisy elements (Type 4) are introduced. The noisy test sequences are generated by adding white noise (with zero mean and standard deviation σ) and uniform noise (in the range of $[-U, U]$) to the original sequence.. The values of σ are 0.01, 0.05, 0.1, 0.15, 0.2 and 0.3 which correspond to the test sequences WN 1 to WN 6 in Table 1. The values of U are 0.05, 0.1, 0.15, 0.2, 0.25 and 0.3 which correspond to the test sequences UN1 to UN6. The simulations with noisy sequences were conducted with 1000 random trials for each σ or U. The average outputs with unnormalized (absolute) activations and normalized (absolute values divided by O_{max}) activations are reported. The decay parameter γ is set to 0.2.

The first observation is that the original sequence elicits the maximum activation ($O_{max} = 3.2$) among all the cases. Secondly, for each type of distortion, the output of the LTM cell reflects an increase of the distortion level by graceful degradation of activation.

Input	*T*	*NO*	*UO*	*Input*	*T*	*NO*	*UO*
ABCD	0	1.000	3.200	**A**	3	0.250	0.800
ABDC	1	0.750	2.400	**B**	3	0.250	0.800
ACBD	1	0.688	2.200	**WN 1**	4	0.994	3.182
ADBC	1	0.375	1.201	**WN 2**	4	0.855	2.735
CBAD	1	0.500	1.600	**WN 3**	4	0.572	1.830
DCBA	1	0.250	0.800	**WN 4**	4	0.387	1.237
ABBCD	2	0.938	3.000	**WN 5**	4	0.290	0.929
ABCCD	2	0.938	3.000	**WN 6**	4	0.132	0.421
ABBBCD	2	0.875	2.800	**UN 1**	4	0.951	3.042
ABCCCD	2	0.875	2.800	**UN 2**	4	0.808	2.584
ACD	3	0.750	2.400	**UN 3**	4	0.632	2.021
BCD	3	0.750	2.400	**UN 4**	4	0.471	1.506
AB	3	0.500	1.600	**UN 5**	4	0.364	1.166
BC	3	0.500	1.600	**UN 6**	4	0.280	0.895

Table 1: Output of the LTM cell with various distortion of an input sequence (Notations: T: perturbation type, NO: normalized output, UO: unnormalized output)

B. Experiment 2: Classification of hand sign language with ASL dataset.

The ASL dataset contains samples recorded by a high-quality hand position tracker from a native signer expressing various Auslan signs. The total number of signs is 95 words with 3 samples per word per session spanning the period of 9 different sessions. Each sample contains a multi-variate temporal pattern of average length of 57. Each sample contains 22 analog features. The dataset encapsulates many spatio-temporal variations from sources such as sensory noises, manipulation speeds of expression and fatigue of the signer. The task for using this dataset in our experiment is to perform sign classification given a test sample using the proposed LTM model.

In this work, we use the 1st derivatives of the $x - y$ coordinates of both hands (4-dimensions) as the feature set. Additionally, each dimension of the extracted trajectories is pre-processed by a moving average window of size 3. In this work, a similar experimental setup as in [20] was used. We used half of the trajectories (i.e. 13 samples per sign) as the training set and all the available trajectories as the testing set (i.e. 27 samples per sign). To achieve a desirable performance, two parameters needed to be determined, namely the decay rate γ and the significance factor η. For decision making, a test sample was assigned to the sign of the maximum activated LTM cells and a correct prediction was counted if the assigned label is similar to the true label.

We organized two different experiments with the ASL dataset. In this first, we learn each sample of the training set with a separate LTM cells and evaluates the performance with nearest-neighbor (NN) classifier. In the second, we examine a sequence alignment procedure based on the LTM activations' profile of sequences belonging to a similar class.

1) Classification with nearest-neighbor

In this experiment, each of the samples of the training set was stored as a separate LTM cell with the label of the corresponding hand sign. The parameters were optimized by a 3-fold cross validation on the training set in the grid $\gamma \in [0,1]$ (with the grid step of 0.1) and $\eta \in [0,2^0, 2^1 \ldots 2^6]$. We performed experiments with 4 different numbers of selected classes, namely 8, 16, 29 and 38. For a number of classes C, we repeatedly collected samples from C random signs of the total 95 signs for multiple runs.

To quantify the results, we used 3 criteria: Prediction accuracy (PA) of classification, normalized activation (NA) of the winning LTM cells and separation ratio (SR). The criterion PA is defined as the fraction of correct predictions of test sequences (indicated by the strongest responses from LTM cells). The SR is computed as the ratio between the activation of the winning LTM cell to that of the highest activated LTM cell that belongs to a different class. The SR is computed only for a correct classification of a sample.

To elucidate the sensitivity of the proposed LTM model to the different parameters, for a selected number of classes C, we first obtained the optimal parameters (γ^*, η^*) by cross validation. Subsequently, one of the optimal parameters was fixed while the other was varied. The average results for $C = 16$ in 30 different runs were plotted in Fig. 3.

The first observation is that the performance in terms of PA was consistently improved when each of the parameter was incorporated (by setting the parameter to be positive). The improvement of PA with the modulation of significance (i.e.$\eta > 0$) demonstrates that the proposed significance estimation is appropriate in assisting sign language's interpretation. The second observation is that an improvement of PA was obtained when SR was improved except when the decay rate is high (near 1). In this case, a perfect recognition of an element of the sequence results in only a small increase of activation. This results in a weak LTM activation that translates into high classification ambiguity. Empirically the performance of the model saturated in terms of PA when $\eta \geq 8$. Similar observations

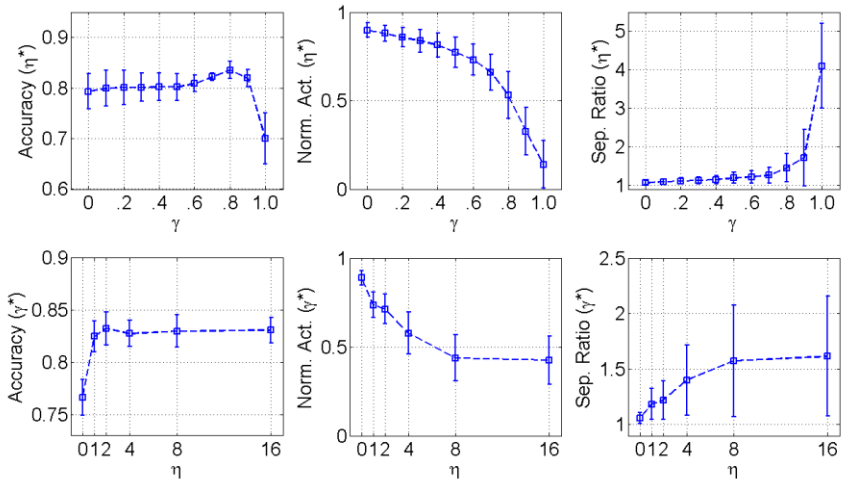

Figure 3: Sensitivity of the LTM cells to the varying parameters for a 16-class classification problem. Top row: PA, NA and SR (from left to right) with varying $\boldsymbol{\gamma}$. Lower row: PA, NA and SR with varying $\boldsymbol{\eta}$. In each row, the other parameter is kept at its optimally found value.

were obtained for different numbers of selected classes.

We benchmark the performance of the LTM model with other published works. Classification accuracy is reported following the protocol in [20]. The performance of the LTM model was compared with the Hidden Markov Model (HMM) and Gaussian Mixtures Model (GMM) for a similar task. The result is tabulated in Table 2. It can be observed that the proposed LTM model significantly outperforms the other learning models in all selected number of classes with a confidence margin. The results for HMM and GMM were taken from [20].

Model		**Number of classes**			
		8	**16**	**29**	**38**
Proposed LTM Model	**Mean**	*87.10*	*82.90*	*81.10*	*78.90*
	StdDev	0.04	0.03	0.02	0.04
HMM [20]		86.00	78.00	69.00	66.00
GMM [20]		85.00	74.00	67.00	64.00

Table 2: Prediction accuracy (%) of the proposed LTM model and comparisons with other models for the same task

2) *Classification with aligned sequences*

In this experiment, we aim to develop a sequence alignment scheme for combining multiple sequences of similar contents. By combining several examples of aligned sequences that represent a similar class of inputs, we can characterize better intra-element tolerance as in (13). This characteristic is useful to reduce the storage burden of one-shot learning when a large number of training sequences are available. Secondly, it is also useful to learn spatio-temporal structures or grammatical rules from multiple sequences. Finally, sequence alignment is a good foundation to automatic chunking, where structure of words and sentences are obtained without prior supervised partition of the input.

To combine two sequences $\mathbf{S^1}$ and $\mathbf{S^2}$, a fundamental task is to find the matching elements between the two sequences. The sequence alignment procedure based on LTM activation is summarized as follows. Firstly, a sequence $\mathbf{S^1}$ is learnt and stored in an LTM cell. Secondly, the matching elements of the second sequence $\mathbf{S^2}$ can be identified by back-tracking the secondary activation's profile induced by the LTM cell after a sweep of the second sequence. The back-tracking procedure is tabulated in Fig. 4.

[Algorithm 2: LTM Sequence Alignment]

Require: $\mathbf{O}, \boldsymbol{\phi}$

Ensure: $\boldsymbol{\mathcal{A}}$

Initialize:

- $r \leftarrow L$
- $c \leftarrow T$
- $\boldsymbol{\mathcal{A}} \leftarrow \emptyset$

Start Algorithm:

While $(r > 1)$ and $(c > 1)$ do
  While $(r > 1)$ and $(|\mathbf{O}_{r,c} - \mathbf{O}_{r-1,c}| \leq \epsilon\phi_c)$ do
    $r \leftarrow r - 1$
  End While
  While $(c > 1)$ and $(|\mathbf{O}_{r,c} - \mathbf{O}_{r,c-1}| \leq \epsilon\phi_c)$ do
    $c \leftarrow c - 1$
  End While
  $\boldsymbol{\mathcal{A}} \leftarrow \boldsymbol{\mathcal{A}} \cup \{(r, c)\}$
  $r \leftarrow r - 1$
  $c \leftarrow c - 1$
End While
Return $\boldsymbol{\mathcal{A}}$

[End Algorithm]

Figure 4: LTM Sequence Alignment algorithm

We define the output matrix $\mathbf{O} \in \mathbb{R}^{L \times T}$ as the sequence of secondary activation $y_j^{SN}(t)$ $(j = 1 \dots L, t = 1 \dots T)$ estimated as in (9). The alignment procedure makes use of the elements' significance to trace the marked change of the secondary neurons' activation during the sweep of the sequence $\mathbf{S^2}$. It is noted that an activation change of a secondary neuron corresponds to the detection of a respective element of the LTM cell in the input sequence. The following algorithm produces the alignment $\boldsymbol{\mathcal{A}} \equiv \{(r_i, c_i) | \, 1 \leq r_i < L, 1 \leq c_i \leq T\}$. Each tuple in $\boldsymbol{\mathcal{A}}$ describes a matching pair between the $(r_i)^{th}$ element of the LTM sequence and the $(c_i)^{th}$ element of the input sequence.

From Algorithm 2 shown in Fig. 4, there are cases when an element of $\mathbf{S^1}$ does not have a matched element in $\mathbf{S^2}$ or vice versa. For notational convenience, we denote the corresponding tuple in $\boldsymbol{\mathcal{A}}$ as $(-1, c_i)$ where $r_i = -1$ denotes a non-aligned element of c_i. Finally, the tuples in $\boldsymbol{\mathcal{A}}$ are increasingly sorted according to c_i.

Once the alignment $\boldsymbol{\mathcal{A}}$ is determined, a new sequence $\mathbf{S}$ can be constructed as in Algorithm 3 (Fig. 5). When a match is specified in $\boldsymbol{\mathcal{A}}$, a new element is spawned and learnt by combining the two elements from $\mathbf{S^1}$ and $\mathbf{S^2}$ as in (19). On the other hand, when a match is not specified, the element of $\mathbf{S^2}$ is added to the new sequence as in (20). The combination of the two elements is controlled by the learning rate $n_l \in [0,1]$. In the following, the parameter n_l is set to 0.5.

[Algorithm 3: LTM Sequence Combination]

Require: $\boldsymbol{\mathcal{A}}$, $\mathbf{S^1}$, $\mathbf{S^2}$, n_l

Ensure: $\mathbf{S}$

Initialize:

- $\mathbf{S} \leftarrow \emptyset$

Start Algorithm:

For each tuple (r, c) of $\boldsymbol{\mathcal{A}}$

Create a new element S.

If $(r_i \neq -1)$

$$S \leftarrow n_l S_r^1 + (1 - n_l) S_c^2 \quad (19)$$

Else

$$S \leftarrow S_c^2 \quad (20)$$

End If

Add the element S to $\mathbf{S}$:

$$\mathbf{S} \leftarrow \mathbf{S} \cup S$$

End For

Return $\mathbf{S}$

[End Algorithm]

Figure 5: LTM Sequence Combination algorithm

We learnt a few sequences for each class by combining the sequences from the training set of each class. The combined sequences were used to predict the class of a new test sequence. For a fair comparison, a similar class selection and corresponding training/testing set to the NN classification in section III.B.1 were used. In this preliminary analysis, we did not perform any additional cross-validation to optimize the parameters for the combined sequence and used the parameters (γ^*, η^*) obtained from the section III.B.1 for each class.

We combined the sequences for each class in a sequential manner. Firstly, a random sequence is selected as a *seeding* sequence. Subsequently, the seeding sequence was combined with a new sample when the matching degree between the two exceeds a combination threshold κ. Otherwise, the new sample is selected as a new seeding sequence. This process continues for the rest of the training samples of the class. In this way, a different number of combined sequences are generated for each class. The order of combination of a training set was perturbed multiple times and the average accuracy and standard deviation is reported.

Fig. 6 shows the classification accuracy when the combined sequences are used with variable κ. It can be seen that the accuracy produced by the combined sequences when $\kappa \geq 0.7$ can be maintained or slightly better than the cases when all the sequences are used ($\kappa = 1$).

Fig. 7 shows the normalized number of combined LTM cells after the combination process. The normalized number is calculated as the number of combined LTM cells divided by the total number of training samples. It can be seen that the normalized number of LTM cells reduce gradually when κ is reduced. It is noted that when κ is equal or below 0.3, all the training samples are combined, thus leaving only a single sequence for representation of each class.

Given that we did not perform any optimization on the learning rate and LTM parameters, the presented results are encouraging and can be further improved in future. A few examples of combined sequences are shown in Fig. 8.

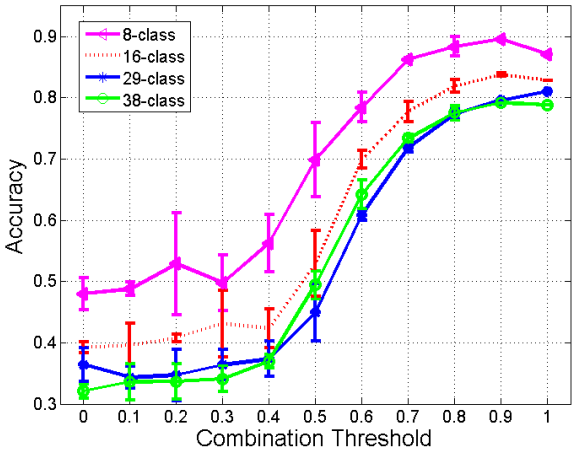

Figure 6: Classification accuracy with variable combination thresholds

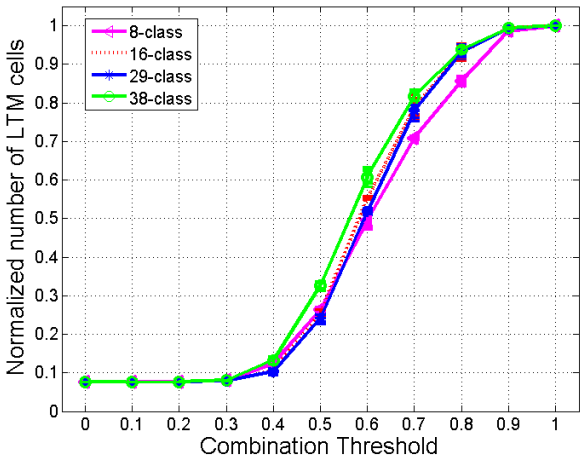

Figure 7: The normalized number of LTM cells with variable combination thresholds

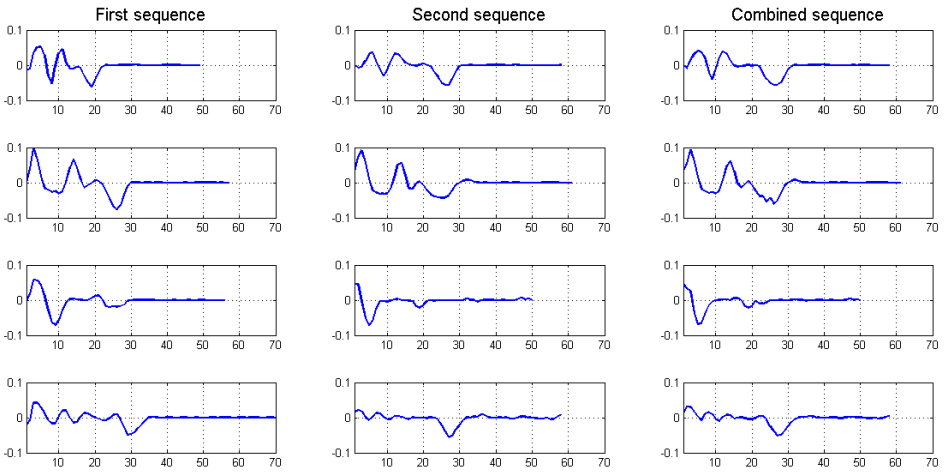

Figure 8: Example of several combined sequences from two arbitrary samples of an ASL word. Only the derivative trajectories of the y-coordinates of the right hand are shown. The words from the top to the bottom row are “science”, “research”, “alive” and “cost”.

IV. Conclusion

In this work, we described a connectionist approach to temporal sequence learning, organization and recognition. The main characteristics of the model include the memory organization of multi-dimensional real-valued sequences, robust matching with error tolerance, significance evaluation of sequence elements and memory forgetting. The merits of the proposed framework were demonstrated by a synthetic example and the ASL dataset. We believe the proposed model is general and can be used with different types of applications that require complex temporal sequence analysis such as speech recognition, robotic navigation, human action recognition and others. Such applications will be explored in future. Several applications of LTM model for text processing and robotic navigation were demonstrated in [3, 21].

The LTM model presented in this paper serves as an essential and constructive element towards the development of machine intelligence. The LTM model aims to construct resilient and stable representation of episodic memory (EM) which is a type of memory that allows one to remember and re-experience previously acquired events and episodes [22]. EM is used for associative memory between sensory inputs and motor actions that are related to machine’s goals and goal creation mechanism [1] in autonomous systems. In addition, the LTM cell can also be useful in organizing procedural memory where sequences of motor steps are stored and activated to perform complex actions.

References

[1] J. A. Starzyk, *Motivation in embodied intelligence*. Vienna, Austria: I-Tech Education and Publishing, Vienna, Austria, 2008.

[2] D. Wang and M. A. Arbib, "Complex temporal sequence learning based on short-term memory," *Proceedings to IEEE,* vol. 78, pp. 1536-1543, 1990.

[3] J. A. Starzyk and H. He, "Spatio-Temporal Memories for Machine Learning: A Long-Term Memory Organization," *IEEE Transaction on Neural Network,* vol. 20, pp. 768-780, May 2009.

[4] M. W. Kadous, "Temporal classification: Extending the classification paradigm to multivariate analysis," PhD, University of New South Wales, Sydney, 2002.

[5] M. I. Jordan, "Serial Order: A parallel distributed processing approach," in *Neural-network Models of Cognition*. vol. 121, ed: Elsevier, 1997, pp. 471-495.

[6] J. L. Elman, "Finding structure in time," *Cognitive Science,* vol. 14, pp. 179-211, 1990.

[7] J. J. Todd and R. Marois, "Capacity limit of visual short-term memory in human posterior parietal cortex," *Nature,* vol. 428, pp. 751-754, 2004.

[8] D. O. Hebb, *The organization of behavior*. New York: Wiley, 1949.

[9] J. L. McGaugh, "Memory - A century of consolidation," *Science,* vol. 287, pp. 248-251, 2000.

[10] G. Bradski, G. A. Carpenter, and S. Grossberg, "STORE working memory networks for storage and recall of arbitrary sequences," *Biological Cybernetics* vol. 71, pp. 469-480, 1994.

[11] K. J. Lang and G. E. Hinton, "The development of the time-delay neural network architecture for speech recognition," in *Technical Report* ed. Pittsburgh, PA, 1988.

[12] S. Hochreiter and J. Schmidhuber, "Long Short-term Memory," *Neural Computation,* vol. 9, pp. 1735-1780, 1997.

[13] D. Wang and M. A. Arbib, "Timing and chunking in processing temporal order," *IEEE Transactions on Systems, Man and Cybernetics,* vol. 23, pp. 993-1009, 1993.

[14] D. Wang and B. Yuwono, "Anticipation-based temporal pattern generation," *IEEE Transactions on Systems, Man and Cybernetics,* vol. 25, pp. 615-628, 1995.

[15] J. A. Starzyk and H. He, "Anticipation-based Temporal Sequences Learning in Hierarchical Structure," *IEEE Transaction on Neural Network,* vol. 18, pp. 344-358, March 2007.

[16] J. Jonides, R. L. Lewis, D. E. Nee, C. A. Lustig, M. G. Berman, and K. S. Moore, "The mind and brain of short-term memory," *Annual Reviews Psychology,* vol. 59, pp. 193-224, 2008.

[17] R. O. Reilly and Y. Munakata, *Computational Explorations in Cognitive Neuroscience: Understanding the mind by simulating the brain*: MIT Press, 2000.

[18] J. Hawkins and S. Blakeslee, *On Intelligence*. New York: Times Book, 2004.

[19] E. Tulving, "Episodic Memory: From mind to brain," *Annual Reviews Psychology,* vol. 53, pp. 1-25, 2002.

[20] F. I. Bashir, A. A. Khokhar, and D.Schonfeld, "Object trajectory-based activity classification and recognition using Hidden Markov Model," *IEEE Transactions on Image Processing,* vol. 16, 2007.

[21] V. A. Nguyen, J. A. Starzyk, and A. L. P. Tay, "Spatio-temporal learning of visual place cells for robotic navigation," presented at the International Joint Conference on Neural Networks, Barcelona, Spain, 2010.

[22] E. Tulving, "Episodic and semantic memory," in *Organization of memory*, E. T. a. W. Donaldson, Ed., ed: New York Academic Press, 1972, pp. 381-403.