2007 International Symposium on Nonlinear Theory and its Applications
NOLTA'07, Vancouver, Canada, September 16-19, 2007

Recalling Complex Sequences of Patterns Using Neurons with Hysteretic Property

Johan Sveholm†, Yoshihiro Hayakawa† and Koji Nakajima†

†Laboratory for Brainware Systems / Laboratory for Nanoelectronics and Spintronics,
Research Institute of Electrical Communication, Tohoku University
2-1-1 Katahira, Aoba-ku, Sendai, 980-8577 JAPAN
Email: johan, asaka, hello@nakajima.riec.tohoku.ac.jp

Abstract—A network based on the Inverse Function Delayed (ID) model which can recall complex temporal sequences of patterns, is proposed. Complex pattern can be dealt with by extending the network, for each main unit, with buffer units that carries the role of a memory. Replacing the cross correlated weight matrix with a matrix computed from a linear separation problem point of view, these pattern can be stored and recalled even if they are highly correlated. It is shown that for a rather big network complex patterns of degree 3, consisting of highly correlated patterns, can be recalled.

1. Introduction

The dynamics of autocorrelation associative memory can be extended to deal with temporal sequences of patterns by storing pairs between consecutive states, making the recall move from pattern to pattern istead of reaching an equilibrium point. Time in form of *temporal order*, hence, plays a big part in temporal sequences of patterns. Another important time aspect is *time duration*. When for example a melody is to be recalled, since half notes are played for one half the duration of the whole note, that should be taken into consideration when dealing with melodies. Here, the focus is layed on temporal order and how patterns that appear more than once in the same sequence can be dealt with by extending input space, from knowledge of only the present state, to include association to memories.

2. Basic Equations

The model in this paper is based on the Hopfield model [1][2], making it a Hopfield-type sequential associative model for complex patterns in continuous time.

2.1. ID model

The ID model, which includes the biologically plausible BVP model [3], was proposed by Nakajima and Hayakawa [4] in 2002. The model has negative resistance characteristics, which gives it interesting abilities. Further, it allows the output function to be of an S-shape, hence it can have hysteresis characteristics. The ID model is expressed by the following differential equations

$$\tau_u \frac{du_i}{dt} = \sum_j^n w_{ij} x_j - u_i, \tag{1}$$

$$\tau_x \frac{dx_i}{dt} = u_i - g(x_i), \tag{2}$$

where τ_u and τ_x are time constants ($\tau_x << \tau_u$), u_i and x_i is the inner potential and the neuron output, respectively, w_{ij} is the connection strength from the jth neuron to the ith one and typically $g(x) = \frac{1}{\beta} arctanh(x) - Kx$, where β and K are constants.

Note that the dynamics of the ID model are equal to those for the conventional dynamics of the Hopfield model [2] if $\tau_x \to 0$ in Eq. (2). Further, the ID neuron can be compared to a two-state hysteretic neuron [5] when $\beta \to \infty$ and $\tau_x \to 0$. Calling this neuron the *limit ID (LID) neuron*, here is its equations

$$\tau_u \frac{du_i}{dt} = \sum_j^n w_{ij} x_j - u_i, \tag{3}$$

$$x(t) = \begin{cases} -1 & u(t) < -K \\ 1 & u(t) > K \\ nochange & otherwise \end{cases} \tag{4}$$

Hysteretic properties have previously been implemented in the autocorrelation associative memory model [5] and the LID neuron itself has shown to improve the performance on optimization problems, namely the *N-queen problem* and the *4-colouring problem* [6]. With the two-state neuron hysteretic transfer function [7] the behaviour of the network lies very close to the discrete model, in that sense that the two state neuron suddenly makes a switch of output. Hence, the capacity in the continuous time case can be as high as in discrete time. The model in question, is constructed by LID neurons and conventional neurons *(CN)* connected pairwise and defined as the *LIDN-CN model*, giving the equations

$$\tau_c \frac{du_i^c}{dt} = sgn\left(\sum_j^n w_{ij} x_j\right) - u_i^c, \tag{5}$$

$$x_i^c = tanh(B \cdot u_i^c), \tag{6}$$

$$\tau_u \frac{du_i}{dt} = x_i^c - u_i, \tag{7}$$

$$x(t) = \begin{cases} -1 & u(t) < -K \\ 1 & u(t) > K \\ nochange & otherwise \end{cases} \tag{8}$$

where u_i^c and x_i^c are the inner potential and the output, respectively, of the conventional neuron, τ_c a time constant and $B >> 1$.

The weighted external input sum varies for each neuron, causing them to approach the switching of states asynchronously. It is imported to make this approach synchronised, hence, the sign function can be used, making the flow for each LID neuron coincide in strength. Realising this function with a continuous neuron (Eqs. (5) and (6)), however, another feature of great importance is added - delay, τ_c, which is an effective tool when it comes to dealing whith distortion in the network. Adding initial distortion to the inner potential of the LID neuron, the neurons are initally unsynchronised. τ_c is then used to created a time frame in which the neurons can switch, however unsyncronised, and still making the network to be able to reach the next pattern in the sequence [7]. Without the conventional neuron paired with the LID neuron, temporal sequences of patterns cannot be recalled.

2.2. Weight matrix

A sequence of m patterns, $\vec{\xi}^1 - \vec{\xi}^2 - \ldots - \vec{\xi}^m$, is typically stored in the connection weights of a fully connected network, where the pattern vector $\vec{\xi}^\mu$ is pattern μ, and $\vec{\xi}^\mu = \left(\xi_1^\mu, \ldots, \xi_n^\mu\right)^T \in \{-1, 1\}^n$, where n is the total number of neurons and where superscript T denotes the transpose. Further, the sequence is cyclic so that the first pattern follows the last, $\vec{\xi}^{m+1} = \vec{\xi}^1$.

The shortest subsequence of patterns that determines $\vec{\xi}^\mu$ is called the *degree*, g, of $\vec{\xi}^\mu$ [11]. Hence, the degree of the entire sequence is the maximum degree of its individual components. For a *simple sequence*, where each pattern is unique, the transition from one pattern to its successor is decided by the present state of the network and is thus classified as degree 1. However, to recall a sequence that contains duplicated patterns, knowledge of the present state alone is not longer sufficient, additional associations to previous memory states, *a short-term memory*, have to be implemented. Such a sequence is of degree greater than 1 and is defined as a *complex sequence* [11].

Storing the sequence in a cross-correlation matrix [1] lies very close to what would result from iteration by the Hebb rule [8]. However, since difficulties arise when dealing with non-random patterns that have a lot of features in common (letters of the alphabet etc.), the weight matrix W is computed from a linear separation problem point of view, that guarantees the storage [9]. The equation

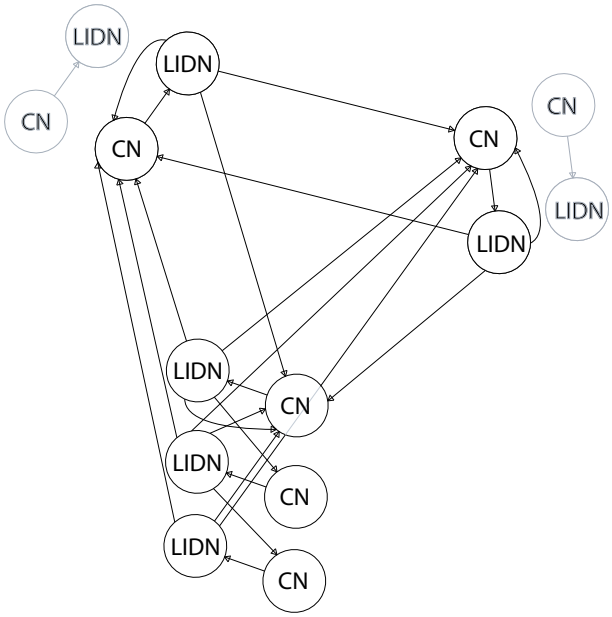

Figure 1: The LIDN-CN network for recall of complex temporal sequences of patterns. Three main units and four buffer units (two shaded) are plotted. The conventional neuron of the main unit has connections from all other LID neurons (main and buffer). The main unit at the bottom of the figure has two buffers unit serially connected. The buffer units are not connected to any other main units' buffer units.

$$W\vec{\xi}^\mu = \vec{\xi}^{\mu+1}, \tag{9}$$

where $\mu = 1, \ldots, m$, would in case of complex patterns take the form

$$W^1\vec{\xi}^\mu + W^2\vec{\xi}^{\mu-1} + \ldots + W^g\vec{\xi}^{\mu+g-1} = \vec{\xi}^{\mu+1}. \tag{10}$$

From here, the behaviour of each neuron through the sequence is considered as a separate problem, each creating a matrix of dimension $m \times (n \times g + 1)$. These matrices are converted into reduced row-echelon form, from which the weight matrices, $W^1 \ldots W^g$, can be computed row by row.

2.3. Complex sequences

A short-term memory model stores information about the past components of a given input sequence, which allows the network to establish a temporal association between consecutive patterns in a sequence. A simple way of defining the dynamics in discrete time is by [10]

$$x_i(k+1) = \sum_j^n w_{ij}^1 x_j(k) + \sum_j^n w_{ij}^2 x_j(k-1) + \ldots + \sum_j^n w_{ij}^g x_j(k-g-1). \tag{11}$$

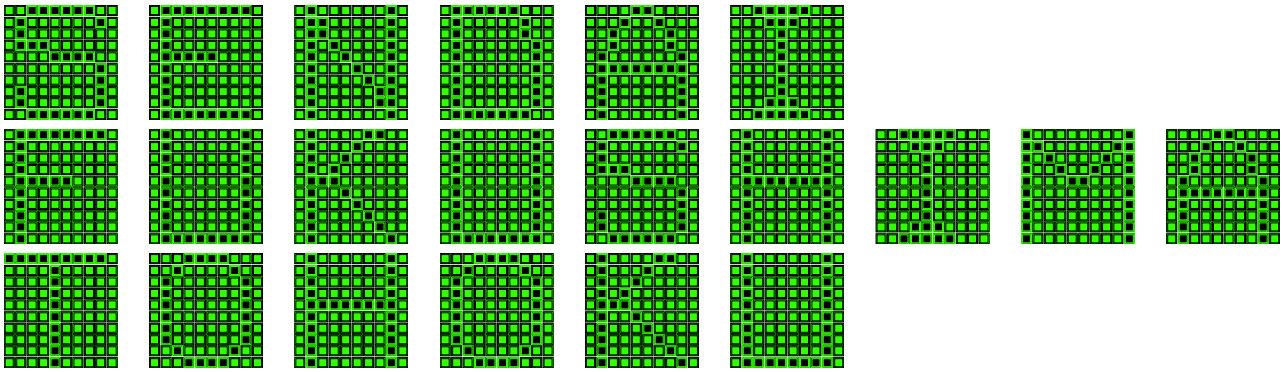

Figure 2: The complex sequence of patterns used in the example, spelled out S-E-N-D-A-I, F-U-K-U-S-H-I-M-A, T-O-H-O-K-U. The network is 10×10 neurons in size, with three sequences of total 21 patterns.

One can create a simple short-time memory, by using a buffer of g units that stores g past states. In discrete time, these are updated each time step [11], in continuous time, however, one have to depend on time delay. In the model presented here, each main unit is not only fully connected to each other main unit, they are also connected to each buffer unit, as seen in Fig. 1. The buffer units themselves are serially linked one after another from the main unit they are buffering, but are not connected to other main units' buffer units.

For a network of $g = 2$, the main units operates according to the equations

$$\tau_c \frac{du_i^{c1}}{dt} = sgn\left(\sum_j^n w_{ij}^1 x_j^1 + \sum_j^n w_{ij}^2 x_j^2\right) - u_i^{c1}, \tag{12}$$

$$x_i^{c1} = tanh(B_1 \cdot u_i^{c1}), \tag{13}$$

$$\tau_u \frac{du_i^1}{dt} = x_i^{c1} - u_i^1, \tag{14}$$

$$x_i^1 = sgn\left(u_i^1 + K_1 x_i^1\right) \tag{15}$$

and the buffer units, delayed one pattern in time, according to

$$\tau_c \frac{du_i^{c2}}{dt} = x_i^1 - u_i^{c2}, \tag{16}$$

$$x_i^{c2} = tanh(B_2 \cdot u_i^{c2}), \tag{17}$$

$$\tau_u \frac{du_i^2}{dt} = x_i^{c2} - u_i^2, \tag{18}$$

$$x_i^2 = sgn\left(u_i^2 + K_2 x_i^2\right). \tag{19}$$

A network of $g = 3$ only adds another buffer unit.

3. Results

Visual images of the letters in the alphabet are stored consisting of 10×10 neurons. In the example Japanese names from the Miyagi prefecture are chosen *SENDAI*, *FUKUSHIMA*, *TOHOKU*. The patterns can be seen in Fig. 2. Adding these words one by one for recall of the letters in a sequence, a network with degree 1, 2 and 3, repspectively, is in need, according to

- $degree(SENDAI) = max(1, 1, 1, 1, 1, 1) = 1$
- $degree(SENDAI|FUKUSHIMA)$
  $= max(2, 1, 1, 1, 2, 2|1, 2, 1, 2, 2, 1, 2, 1, 2) = 2$
- $degree(SENDAI|FUKUSHIMA|TOHOKU)$
  $= max(2, 1, 1, 1, 2, 2|1, 2, 2, 3, 2, 2, 2, 1, 2|1, 2, 2, 2, 2, 3)$
  $= 3$

In order to recall a sequence of higher degree for a rather large network, all the neurons have to be initialised from the start i.e. not only the starting pattern (letter) is presented to the main units, but also previoius patterns have to be presented for the buffer units. After that, recall works smothly, each buffer unit *remembering* one state back in time when the network goes through the sequence. The state transition of the network can be seen in Fig. 3.

4. Conclusion

Two things were especially important in order to complete the presented model.

Implement short-term memory. In order to recall complex sequences, in where patterns appear more than once, some kind of short-term memory needs to be implemented. In discrete time this can easily be made with a buffer of g units, that stores the most recent g inputs. In continuous time, however the time step has to be replaced with time delay. This was made possible with buffer units that feed past states to the main units with the same time delay it

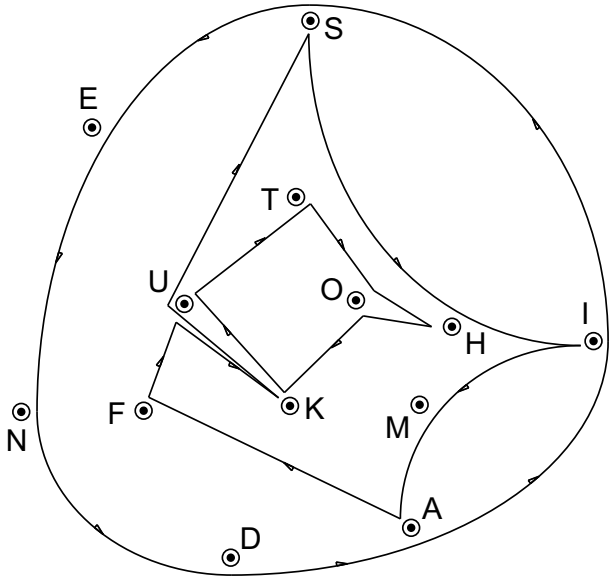

Figure 3: State transition of the network, where each dot (letter) represents one state of the network: *SENDAI, FUKUSHIMA, TOHOKU*.

takes for the network to complete one transition from one pattern to another.

Compute the weight matrix. Storing the temporal order of sequences of patterns in a cross correlations matrix, works well with randomly generated patterns, but not when dealing with patterns that are highly correlated. One way to deal with this problem is to preprocess the input to the network, so that the sequence of patterns used in the actual network contains a simulated randomness. For the observer to understand content of the network, however, the output has to go through the reverse process of the input manipulation.

We chose another method, in where the network operates with unmanipulated data, but with a weight matrix that, instead of being based on the concept of correlation, is calculated to guarantee the storage of data.

The present network needs to have both main units and buffer units initialised, but future studies aim to decrease the dependence of initialised buffer units. It is instead shown that the present model can recall highly correlated complex patterns of rather high dimension (=100) and degree (=3).

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA*, vol.79, pp.2254–2558, 1982.

[2] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proc. Natl. Acad. Sci. USA*, vol.81, pp.3088–3092, 1984.

[3] R. Fitzhugh, "Impulses and Physiological States in Theoretical Models of Nerve Membrane," *Biophysical J.*, vol.2, pp.445–466, 1961.

[4] K. Nakajima, Y. Hayakawa, "Characteristics of Inverse Delayed Model for Neural Computation," *Proc. of NOLTA 2002*, pp.861–864, 2002.

[5] H. Yanai, Y. Sawada, "Associative Memory Network Composed of Neurons with Hysteretic Property," *Neural Networks*, vol.3, pp.223–228, 1990.

[6] A. Sato, Y. Hayakawa, K. Nakajima, "The Parameter Dependence of the Inverse Function Delayed Model on the Success Rate of Combinatorial Optimization Problems," *IEICE Trans. Fundamentals*, vol.J89-A, no.11, pp.960–972, 2006.

[7] J. Sveholm, Y. Hayakawa, K. Nakajima, "Recalling Temporal Sequences of Patterns Using Neurons with Hysteretic Property," *Proceedings of the 20th Workshop on Circuits and Systems in Karuizawa*, 2007, in print

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

[9] I. Guyon, L. Personnaz, J. P. Nadal, G. Dreyfus, "Storage and retrieval of complex sequences in neural networks," *Physical Review A*, vol.38, no.12, pp.6365–6372, 1988.

[10] A. Goulon-Sigwalt-Abram, A. Duprat, G. Dreyfus, "From Hopfield nets to recursive networks to graph machines: Numerical machine learning for structured data," *Theoretical Computer Science.*, vol.344, pp.298–334, 2005.

[11] D. Wang, B. Yuwono, "Anticipation -Based Temporal Pattern Generation," *IEEE Trans. on Sys., Man and Cybern.*, vol.25, no.4, pp.615–628, 1995.