A Neural Network for Temporal Pattern Recognition

T.H. Yeap, S.G. Zaky, J.K. Tsotsos and H.C. Kwan

Departments of Electrical Engineering, Computer Science and Physiology
University of Toronto, Ontario, Canada

ABSTRACT

A neural network that can be trained to recognize a sequence of events in time is presented. It utilizes a process of temporal integration, which is implemented by providing each node with a fading memory and the use of a special interconnect pattern. A simulation model for a motion detector based on the proposed network was implemented. It can be trained to distinguish between several speeds of a moving pattern of random dots.

1. Introduction

A neural network is a massively parallel machine comprising many simple processing nodes that are totally or partially connected to each other by weighted links. It can be programmed by adjusting the weights of the links. Recent work on neural networks has concentrated on their ability to store, retrieve and process static patterns [Hopfield 79, Hinton 84, Ackley 86, Rumelhart 86]. However, in problems such as motion detection, speech processing and predictive signal processing, the information to be processed consists of a sequence of temporally related patterns. Neural networks that handle only static patterns are not suited for these problems.

Several approaches have been proposed for the use of neural networks to recognize a temporal sequence [Grossberg 67, Grossberg 72, Kohonen 84, Mozer 88]. One approach involves the use of buffers and time delays [Tank 87]. This technique is based on transforming a temporal problem into a static pattern. Time is not represented directly. In computer vision, a framework that integrates time into high-level vision can be found in ALVEN — an expert system that evaluates the performance of the human left ventricle from a sequence of X-ray images [Tsotsos].

This paper introduces a neural network that utilizes the process of temporal integration to recognize a sequence of events in time. The time element is introduced by using nodes with a fading memory. Each node is associated with one event of the sequence. When the network recognizes one of these events, it sends a positive excitation signal to the corresponding node. However, this node will only be partially activated, unless the node representing the previous event in the sequence is still active. A complete sequence of events is recognized by temporally integrating the activity of the nodes representing the events of the sequence. By using an appropriate threshold, the number of events of the sequence that must be detected by the network before a decision is made can be controlled. Due to the temporal integration process, a short burst of noise appearing at the input of the network will not undo the results of many successful samples during the recognition process.

The design of the proposed network has been tested by simulating its use to implement a motion detector. Using a moving random dot pattern as the test input, the network has been shown to be capable of recognizing a variety of input sequences resulting from regular and irregular motion.

The proposed network is modular in architecture, and is designed to be suitable for implementation using current VLSI technology. It consists of several temporal modules, all connected to the same external inputs. Each module is responsible for recognizing a particular sequence of events. For example, in the case of motion detection, different modules may detect motion at different speeds, or some modules may recognize particular patterns of irregular motion.

Section 2 below describes the architecture of the temporal module and how several modules may be interconnected for multiple sequence detection. The motion detection application is described in Section 3, and simulation results are presented in Section 4.

2. Architecture of a Temporal Module

A temporal module comprises N *Sequence* nodes ($S_1..S_N$) and a *Result* node (R_T) interconnected as shown in Figure 1. Excitatory connections from sequence nodes S_1 to S_2, S_2 to S_3, etc. form a forward chain. In addition, each sequence node receives inhibitory inputs from all other nodes except its upstream neighbor. Output from each sequence node is connected to the input of the result node by a fixed weight excitatory connection. When a network has more than one temporal module, the result node of each module receives inhibitory inputs from the result nodes of other modules.

For any network to recognize a sequence of events in time, it must be able to store information, at least temporarily, about individual event of the sequence. In a temporal module, the time element is introduced by using sequence nodes with fading memory. As a result, when a node is activated in response to a particular input event, it remains active for a short period after the input stimulus has been removed. By remaining active, it helps the node downstream to fire when the next pattern in the sequence is presented. Thus, the direction of time flow is kept track of by using the excitatory connections in the figure. Input events are assumed to be presented to the machine at regular intervals.

A node is partially activated in response to its corresponding event, even without encouragement from other nodes. To ensure that only correct sequences are recognized, each node receives inhibitory signals from all other

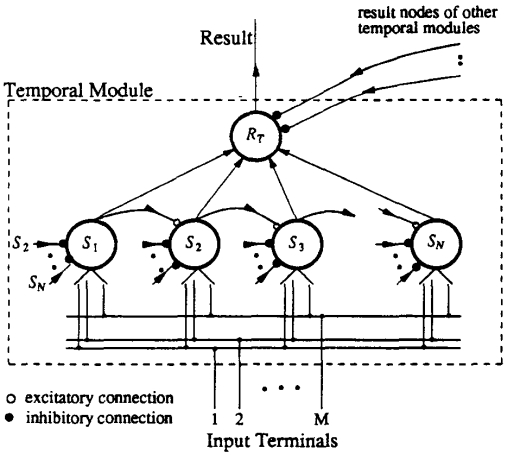

Fig. 1 A temporal module of nodes

nodes except its upstream neighbor. The excitatory and inhibitory connections, together with the ability to partially remember previous history, enable a chain of sequence nodes to recognize a temporal sequence appearing at the external inputs. A correct input sequence causes the nodes to fire from left to right.

When a correct sequence is presented to a temporal module, there will always be at least one sequence node active. This can be guaranteed by arranging for the time constant of the fading memory to be larger than the inter-arrival time of the input events. Each active sequence node sends an excitation signal to the result node, R_T, via a fixed-weight excitatory connection. The result node integrates the sum of these signals with respect to time, and it becomes active when the integral exceeds a predefined threshold. The integration time constant together with the value of the threshold determine the number of correct input events that need to be recognized before the result node fires.

Circuit Implementation

The nodes in a temporal module may be implemented using the circuit shown in Figure 2. Each node comprises an operational amplifier with a capacitor and a resistor connecting the input of the amplifier to ground. The input of the amplifier is also connected to the outputs of other nodes via voltage-dependent current sources, which implement the necessary excitatory or inhibitory connections between the nodes. In the simulation model used to test the operation of the temporal modules, the operational amplifier in each node is assumed to have a sigmoidal transfer function of the form:

$$V = \frac{V_{dd}}{1 + e^{\frac{-U}{A}}}$$

where A is the gain of the amplifier, Vdd is the supply voltage, and U and V are the input and output voltages respectively. A sigmoidal function is commonly used in neural network studies because it closely models the activity of a neuron [Hopfield 88]. It is used here because it also constitutes a reasonable model for an operational amplifier when saturation is taken into account. Moreover, its derivatives are continuous and easily computed.

The capacitor and resistor at the input of each amplifier provide the fading memory, whose time constant is given by $\tau = C / G$. The weight of the connection from the output of node j to the input of node i is defined by the mutual conductance g_{mij} of the current source connecting the two nodes. The mutual conductance will be regarded as positive for excitatory connections and negative for inhibitory connections.

Applying Kirchoff's current Law at the input of the amplifier in each of the sequence nodes in Figure 1, we obtain:

$$\tau \frac{dU_i}{dt} + U_i = \frac{1}{G}\left(\sum_{j=1, j \neq i}^{N} V_j g_{mij} + \sum_{j=1}^{M} E_j g_{ij} + V_{dd} g_{bi} \right)$$

where E_j, $j = 1 \ldots M$, is the external input voltage.

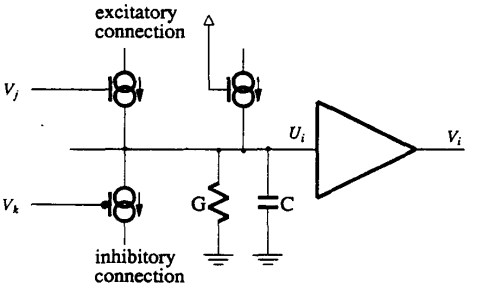

Fig. 2 Circuit Model of a Node

Training

A temporal module can be trained to recognize a given sequence of events using the least-mean-square error method [Tou]. For each event in the sequence, the interconnection weights should be adjusted such that only the corresponding sequence node becomes active, i.e. having $V_i = V_{dd}$. In each step of the iterative training process, the mutual conductances of the current sources are updated based on the errors in the response of the sequence node. Weights are increased slightly where that would lead to a result that is more consistent with the desired response. Otherwise they are decreased.

For a given input event, only the weights associated with the corresponding sequence node are adjusted. That is the machine is trained to recognize the input events independently of each other. Training is completed for one event, then for another, etc. in any order. However, while training the machine for event i, which should activate sequence node i, node $i-1$ is assumed to be active, and all other nodes inactive.

3. The Motion Detector

Given a sequence of images of an object moving in a plane parallel to the image plane, the function of a motion detector is to estimate quantities such as the speed and direction of the movement. Significant research effort has been spent on this problem, but the issue of accounting for the passage of time has not been adequately addressed [Tsotsos]. In the motion detector described below, time is incorporated directly into the computation by using the temporal integration capability of the modules described above.

(217) 333-4789

Further information can be obtained from Dr. W. Kenneth Jenkins.

The architecture of the motion detector is shown in Figure 3. It consists of three layers: input sampling, encoding and sequence recognition. The input sampling layer samples the image input at a fixed rate. A discrete time differentiator, which generates an output of one if the present input is different from its previous value and zero otherwise, is provided for each pixel of the image. A moving edge causes a change in brightness at some pixels, hence a non-zero temporal derivative.

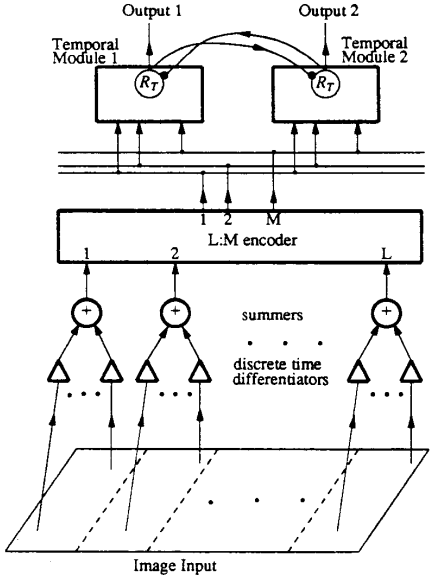

Fig. 3 Architecture of a Motion Detector

The magnitude of the time derivative of the signal from each pixel in a small region of the image is summed by a linear summer. Thus, the summer's output will be between 0 or V_{dd} volts, depending on the density of pixels which have changed in brightness during a sampling period. The outputs of the summers are connected to an encoder before being fed to the sequence recognition layer. The encoder reduces the number of inputs that have to be processed by the sequence recognition layer.

The sequence recognition layer consist of several temporal modules, each responsible for recognition of a specific type of motion. The result node of each module receives inhibitory signals from all other modules. Thus, the module producing the strongest response will inhibit the other modules.

4. Simulation Results

The architecture of the motion detector described in Section 3 has been simulated and tested. Two motion detectors have been implemented, one for single speed and one for multi-speed one-dimensional motion. This section presents some of the simulation results.

There are two types of image input that can be used in computer vision to quantify the performance of a motion detector: a sinusoidal grating and random dots [Adelson 86]. For ease of interfacing with the motion detector, the image input used was a random dot pattern moving against a background containing a different random dot pattern.

The simulation results for the single-speed motion detector is given in Figures 4a and 4b. Two temporal modules were used. Module 1 was trained to recognize an object that is moving left at a speed of 8 pixels per unit time. Module 2 was trained to recognize motion in the opposite direction. The input image was sampled once every unit time. Figure 4a shows the response of both modules when the stimulus is moving right at a speed of 5 pixels per unit time. Module 2 begins to fire at about the 8th. sample, while module 1 remains inactive. When the direction of motion is reversed, the response of the modules is reversed, as seen in part (b) of the figure. Figure 5 shows the performance of the motion detector for different speeds. For a training speed, S_T, of 8 pixels per unit time, the modules recognize objects moving at speeds in the range 2-10 pixels per unit time.

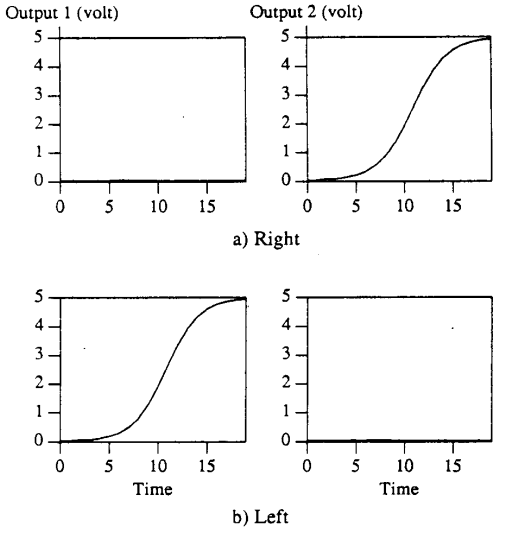

Fig. 4 Response of Motion Detector to a moving object

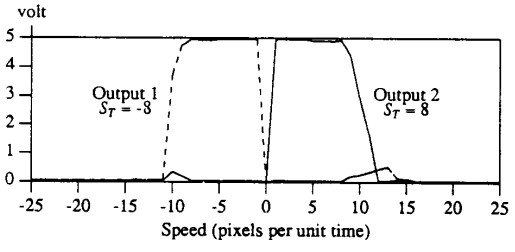

Fig. 5 Output of Motion Detector trained at 2 speeds

The case of a motion detector containing 6 modules trained at 6 different speeds is shown in Figure 6. The effect of the inhibitory interconnections between modules can be clearly seen in the sharp changes in response at the intermediate speeds of –10, 0 and 10 pixels per unit time.

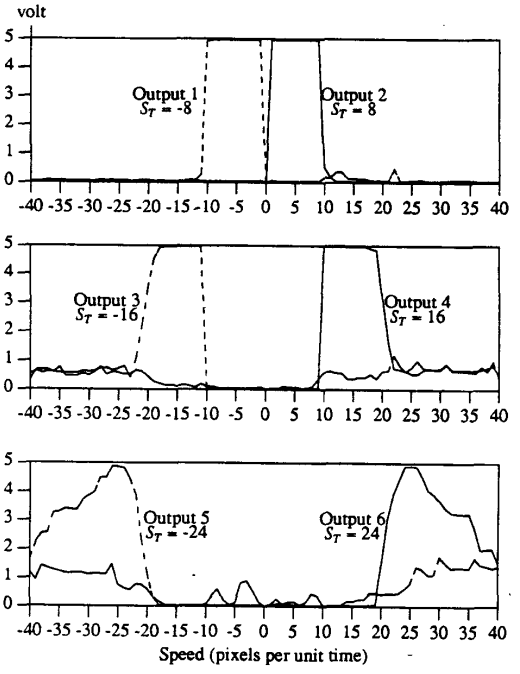

Fig.6 Outputs of Motion Detector trained at 6 speeds

Simulation also showed that the temporal module is capable of detecting irregular motion. The random dot pattern was moved to the right at a fixed speed for a short period; then, the direction of motion was reversed. Work on this aspect, as well as the ability to recognize motion in two dimensions, is continuing.

5. Conclusion

The structure of a temporal module that utilizes temporal integration to perform sequential pattern recognition has been described. Simulation results show that a module based on chaining of excitatory connections between nodes with fading memory can be used for recognizing time-varying inputs. A motion detector using several temporal modules has been shown to be capable of recognizing a variety of temporal sequences resulting from motion at different speeds.

6. References

1. Ackley D.H., Hinton G.E., Sejnowski T.J., "A Learning Algorithm for Boltzmann Machine", *Cognitive Science*, Vol. 9, 1985, pp. 147-169.
2. Adelson E.H., Bergen J.R., "The extraction of Spatio-temporal Energy in Human and Machine Vision", *I.E.E.E. Proc. Workshop on Motion: Representation and Analysis*, May 1986, pp. 151-155.
3. Grossberg S., "Some Networks That Can Learn, Remember and Reproduce Any Number of Complicated Space-Time Patterns, I", *Journal of Mathematics and Mechanics*, Vol. 19, 1969, pp. 53-91.
4. Grossberg S., "Some Networks That Can Learn, Remember and Reproduce Any Number of Complicated Space-Time Patterns, II", *Studies in Applied Mathematics*, Vol. 49, 1970, pp. 135-136.
5. Hinton G.E., Sejnowski T.J., Ackley D.H., "Boltzmann Machines: Constraint Satisfaction Networks that Learn", Technical Report CMU-CS-84-119, Carnegie-Mellon University, May, 1984.
6. Hopfield J.J., "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", *Proceedings of the National Academy of Sciences USA*, 79, pp. 2554-2558.
7. Hopfield J.J., "Artificial Neural Network", *I.E.E.E. Circuits and Devices Magazine*, Sept. 1988.
8. Kohonen T., *Self-Organization and Associative Memory*, Springer, Berlin, 1984, pp. 16-20.
9. McCelland D.E., Rumelhart D.E., *Parallel Distributed Processing - Vol. 1 & Vol. 2*, The MIT Press.
10. Mozer M.C., "A Focused Back-Propagation Algorithm for Temporal Pattern Recognition", Technical Report CRG-TR-88-3, June 88, University of Toronto.
11. Rumelhart D.E., Hinton G.E., Williams R.J., "Learning Representation by Back-Propagation Errors", *Nature*, Vol. 323-9, October 1986, pp. 533-536.
12. Tank D.W., Hopfield J.J., "Neural Computation by Concentrating Information in Time", *Proc. Natural Academy of Science, USA*, Vol. 84, April 1987, pp. 1896-1900.
13. Tou J.T., Gonzalez R.C., *Pattern Recognition Principles*, Addison-Wesley Publishing Company.
14. Tsotsos J.K., "Representational Axes and Temporal Cooperative Processes", *Vision, Brain and Cooperative Computation*, Bradford Book, MIT Press, pp. 361-417.

(217) 333-4789 Further information can be obtained from Dr. W. Kenneth Jenkins.