ELSEVIER

Cognitive Systems Research 3 (2002) 255–270

Cognitive Systems RESEARCH

www.elsevier.com/locate/cogsys

Hybrid preference machines based on inspiration from neuroscience

Action editor: Franz Kurfess

Stefan Wermter*, Christo Panchev

University of Sunderland, Informatics Centre, SCET, St. Peter's Way, Sunderland SR6 0DD, UK

Received 15 December 2000; accepted 17 May 2001

Abstract

In the past, a variety of computational problems have been tackled with different connectionist network approaches. However, very little research has been done on a framework which connects neuroscience-inspired models with connectionist models and higher level symbolic processing. In this paper, we outline a preference machine framework which focuses on a hybrid integration of various neural and symbolic techniques in order to address how we may process higher level concepts based on concepts from neuroscience. It is a first hybrid framework which allows a link between spiking neural networks, connectionist preference machines and symbolic finite state machines. Furthermore, we present an example experiment on interpreting a neuroscience-inspired network by using preferences which may be connected to connectionist or symbolic interpretations.

Keywords: Hybrid systems; Neural preferences; Preference machines; Neural networks of spiking neurons

1. Introduction

Recently, there has been some preliminary work integrating principles from neuroscience into computational models e.g. (Maass & Bishop, 1999; Thorpe, Fize, & Marlot, 1996; Wermter, Austin, & Willshaw, 1999b; Taylor, 1999; Denham & Denham, 1999). Although neuroscience principles have helped to develop new computational models, the problems they address are still restricted. In many ways there is a challenging distance between lower cognitive neuroscience and higher structural concepts. However, long-term progress needs cognitive science and neuroscience to be taken more seriously by computer scientists for making progress in high-level processes like language understanding.

Our approach attempts to go beyond the existing connectionist approaches that are normally utilized (Rumelhart, Hinton, & Williams, 1986; Feldman & Ballard, 1982). Since 2000, the computational neuroscience network EmerNet has explored emerging computational neural network architectures based on neuroscience (Wermter, Austin, & Wilshaw, 1999b;

*Tel.: +44-191-515-3279; fax: +44-191-515-3553.
E-mail address: stefan.wermter@sunderland.ac.uk (S. Wermter).

1389-0417/02/$ – see front matter © 2002 Published by Elsevier Science B.V.
PII: S1389-0417(01)00061-4

Wermter, Austin, & Willshaw, 2001) (http://www.his.sunderland.ac.uk/emernet). This paper is based on this context and attempts to outline a first hybrid framework based on neuroscience, in particular for language processing. Since the processing of sequential patterns and preferences are an inherent property of language, we will develop this framework around different forms of sequential machines at symbolic, connectionist and neuroscience levels. Our approach is inspired from the processing in the brain, integrates sequential machines at diverse levels, both vertically and horizontally, and exploits recurrent and pulsed neural networks for more neuron-like processing.

Our overall preference machine framework (Fig. 1) can be summarized as follows. At the highest level, there are symbolic structures, e.g. based on symbolic machines like Moore machines (Hopcroft & Ullman, 1979) and most such representations lack graded preferences. At the middle level, connectionist preference machines have graded preferences but lack detailed temporal processing (Wermter, 2000b). At the neuroscience-inspired level, dynamic processing and preferences exist and provide the potential for temporal processing (Panchev & Wermter, 2000).

A Preference Machine is a core element of this framework. It can be seen as a computational machine which has possible links to higher level symbolic machines or lower level neuroscience-inspired concepts. For instance, a connectionist preference machine (Elman, Bates, & Johnson, 1996) can be interpreted symbolically as a finite state machine. It has been shown (Wermter, 2000a), that symbolic machines can be extracted from SRNs using our preference framework. Each state and each output within this preference Moore machine was mapped towards the references of an n-dimensional space. That way, a symbolic machine represented a higher, more abstract representation of the more detailed connectionist preference Moore machine. On the other hand, it has been demonstrated (Shavlik, 1994; Towell & Shavlik, 1994; Omlin & Giles, 1996b) that symbolic automata can be transformed for inducing connectionist sequential machines.

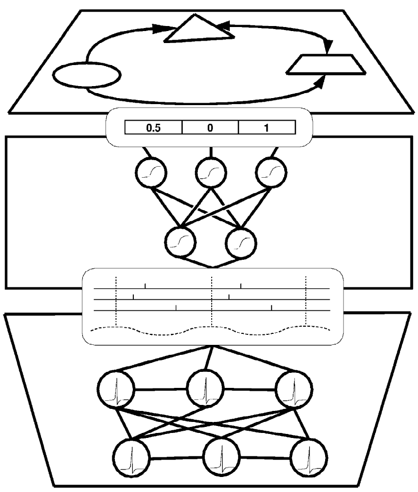

Fig. 1. General framework of connecting neuroscience-inspired, connectionist and symbolic computation.

The focus of this new approach here is to explore hybrid neural architectures, including techniques from cognitive neuroscience and neural computation in order to produce computational neural models of language processes and complex cognitive operations. These models can also create general notions on language and the brain, and identify the information requirements for extended models. So, the scope of this paper is not about biological neural networks or neurobiological modeling. Our goal is rather to extend the scope of hybrid approaches from symbolic machines over connectionist machines towards neuroscience-inspired machines.

A substantial part of the information being processed in artificial and biological neural networks is encoded in a distributed manner and is transferred, or sometimes temporally stored, as pulsed signals between neurons. Neurons fire within a given time window, indicating activity with the density or with the particular temporal location of the spikes. Reading such information from real systems or manipulating it in artificial systems is a complex task that addresses many processing and representational problems. In previous work we have introduced preference-based processing (Wermter, 1999, 2000b) and an interpretation of firing rate and pulse coding schemes (Panchev & Wermter, 2000). Here we would like to extend this work substantially towards more complex neural network representations of simple cognitive events.

2. Preference machines

An important aspect of our work is that we want to ground incremental left-to-right processing of language in constraints which are known from cognitive and biological neuroscience. A very basic form of computation can be characterized by finite-state machines (Hopcroft & Ullman, 1979). Therefore, we will focus on connecting symbolic structures, connectionist and neuroscience-inspired neural networks by using concepts based on finite-state machines. These concepts are focused around preference machines which are introduced in this section. Basically a synchronous sequential preference machine transforms sequential input preferences to sequential output preferences. These machine preferences can be used to integrate symbolic and neural knowledge (Wermter, 2000b). In contrast to other research work in the area of finite automata and connectionist networks (Manolios & Fanelli, 1994; Omlin & Giles, 1994; Omlin & Giles, 1996a), we do not only model an acceptor which learns to accept a correct input sequence but we are interested in building robust learning preference machines which can produce output.

We start with a definition of preferences. It is a concept consisting of features which are present to various degrees. In the past, a feature has often been associated with mean firing rate of a neuron (McClelland & Rumelhart, 1986). However, there are several other encoding schemes for neural activity which are seen as complementary to the mean firing rate coding (Maass & Bishop, 1999). For instance, the time-to-first spike coding is based on the relative time between a neuron's stimulus and its response. Furthermore, the synchrony coding is based on the synchrony of neurons which fire in a limited time window (Fig. 2). There is evidence that these various codes exist in the brain in parallel (Abott & Sejnowski, 1999; Maass & Bishop, 1999) and this motivates our representation of combined encoding schemes as preferences.

Definition 1. (Complex preference, briefly c-preference) A complex preference of level l is represented by an $l \times m$-dimensional matrix $a \in [0,1]^{l \times m}$.

The special case of a c-preference of level one is called *simple preference*. The level of a c-preference indicates the number of simple preferences represented in it. In (Panchev and Wermter, 2000), we showed that information represented in mean-firing-rate and temporal neural encoding schemes can be interpreted as c-preferences where the simple preference at each level represents a given internal state of the code. Furthermore, we showed that multiple encoding schemes can be integrated and can be simultaneously processed in a c-preference where each level (or several levels) represent a single scheme. In the following sections, we will build on this previous work to interpret complex neural representations as c-preferences of m-dimensional analog vectors in $[0,1]^m$ or preference Moore machines using c-preferences.

Definition 2. (Next corner reference) The next

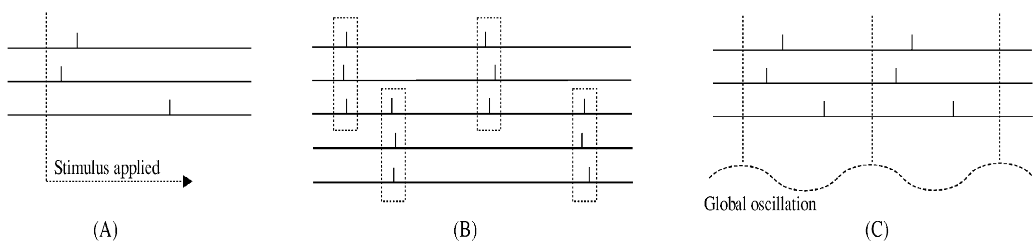

Fig. 2. Examples of temporal neural codes (after Panchev & Wermter, 2000): (A) Time-to-first-spike coding. The neuron in the middle responds faster to the stimulus and therefore indicates the strongest stimulation. The bottom neuron has the weakest response. (B) Synchrony coding. The two neural assemblies (one of the top three neurons and one of the bottom three neurons) can represent two different object/events. (C) Phase coding. The three neurons respond with the phase of the spikes with respect to the periodic background oscillation.

corner reference $r(a) \in \{0,1\}^{l \times m}$ of the c-preference $a \in [0,1]^{l \times m}$ is determined for $i \in \{1, \ldots, l\}$ and $j \in \{1, \ldots, m\}$ as:

$$r_{ij}(a) = \begin{cases} 0 & \text{if } a_{ij} < 0.5 \\ 1 & \text{if } a_{ij} \geq 0.5 \end{cases}$$

The introduction of the next corner reference allows us to associate each c-preference with a particular corner of the $[0,1]^{l \times m}$ hypercube, i.e. a discrete symbolic representation.

Definition 3. (Preference value of a c-preference) A preference value of a c-preference $a \in [0,1]^{l \times m}$ with respect to its next corner reference $r(a)$ is defined as:

$$\text{pref}(a) = 1 - \frac{\text{distance}(a, r(a))}{\frac{\sqrt{lm}}{2}}$$

where

$$\text{distance}(a, r(a)) = \sqrt{\sum_{i,j} (a_{ij} - r_{ij}(a))^2}$$

is the distance between the c-preference a and its next corner reference.

$\sqrt{lm}/2$ is the maximum distance in the $l \times m$-dimensional c-preference space, that is the distance from the center of the hypercube to any corner. If the c-preference a is close to its next corner reference then its preference value pref(a) will be close to 1 and if it is close to the center, then pref(a) will be close to 0.

Definition 4. (c-preference class) Let $a \in [0,1]^{l \times m}$ be a c-preference with next corner reference $r(a) \in \{0,1\}^{l \times m}$. Then the class of complex preferences of a is called c-preference class $c(a)$ and contains all those c-preferences with next corner reference $r(a)$, which have the same distance from $r(a)$ as a.

The *preference value of a class of c-preferences* is the preference value of an arbitrary c-preference which belongs to this class. This follows directly from the definitions of c-preference classes and the preference value. Fig. 3 shows the preference values for the two-dimensional space.

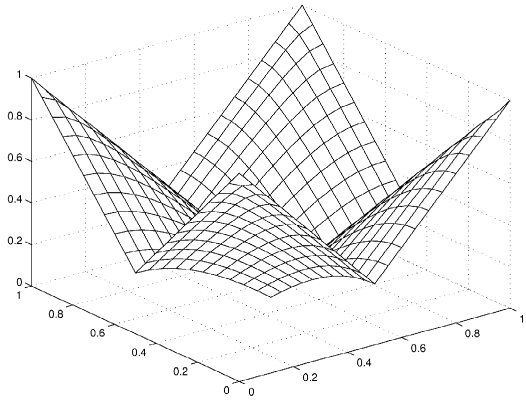

Fig. 3. Preference values z of two-dimensional preferences (x y) (after Wermter, 1999).

A class of preferences represents a high-dimensional hypersphere of an unlimited number of preferences with the same distance from the specified corner reference. Finally, we define a preference machine as a device of sequential processing with c-preferences. For some input and state, a new state and output is computed. Input, output and state are multidimensional preferences.

Definition 5. (Preference machine) A preference machine PM is a synchronous sequential machine which is characterized by a 4-tuple $PM = (I,O,S,f_p)$, with I, O and S being non-empty sets of inputs, outputs and states. $f_p : I \times S \rightarrow O \times S$ is the *sequential preference mapping* and contains the state transition function f_s and the output function f_o. Here I, O and S are n-, m- and l-dimensional preferences with values from $[0,1]^n$, $[0,1]^m$ and $[0,1]^l$, respectively.

In summary, what we have accomplished so far is a preference framework for representing concepts in multiple neural encoding schemes as a $l \times m$-dimensional c-preference. These concepts can be interpreted symbolically based on preference values. Furthermore, these concepts can be integrated with sequential machines. That way Preference Machines are grounded in possibly multiple neural encoding schemes (e.g. mean firing rate coding and time-to-first spike coding) but at the same time they are linked to sequential symbolic machines.

3. Preferences at neuroscience levels

In the remaining part of this paper we will illustrate some links of the preference framework to the neuroscience level. While in (Panchev and Wermter, 2000) we presented the concept of c-preferences on a single neuron level, here we concentrate on more complex cortical functional structures associated with cognitive functions in the brain: cell assemblies and synfire chains.

3.1. Cell assemblies as c-preferences

The concept of a cell assembly was introduced as a functional and structural model for cortical processes and neuronal representations of external events (Hebb, 1949). Hebb presented the idea that complex objects and stimuli, as well as more abstract entities like concepts, ideas and contextual relations in the brain are represented as simultaneous activation of large groups of neurons. Single cells can belong to different assemblies and the cells in one assembly are not necessarily close to each other. If, as a result of an external event, a sufficiently large subset of the cells in the assembly are stimulated, the whole assembly becomes active and may sustain activity for some period of time even when the external event has disappeared.

Cell assemblies are a widely accepted paradigm for feature binding mechanisms in the brain (Shastri, 2001). In many artificial neural networks, cell assemblies are explored as a model of associative memories (Palm, 1986; Fransé, Lansner, & Liljenström, 1992; Pulvermüller, 1999). Different interpretations of the paradigm can serve as a concept of short or long term memory models. The concept of neural assemblies in combination with activity-dependent (spatio-temporal) Hebbian learning provides a paradigm for long term memory (Wennekers & Palm, 1999).

Many artificial neural network models of cell assemblies use a simple neuron as the elementary computational unit of the network. However, there are models of associative memories with spiking neurons that consider cortical columns as the functional units (Fransén & Lansner, 1998; Palm, 1993). Although in both approaches a neuron or column represents a single feature, there are different interpretations of the behavior of that unit. We will focus on two main approaches here outlining the general principles of interpreting the spiking behaviour of cell assemblies as c-preferences. Later on, in Section 4, we will give an example of a more specific interpretation of the assemblies that combines features of the two approaches described here.

3.1.1. Cortical column as a threshold gate

In the first interpretation, a cortical column behaves as a threshold gate, that is, if a sufficient number of excitatory neurons fire, the column is said to be active and the respective feature present. If there are not enough firing neurons, the column is said to be inactive and so is the feature it represents.

Let us consider a model of synchronously firing cell assemblies, with Δt being a time interval in which all spikes would be considered as firing synchronously. A sequence of synchronously firing assemblies will be defined in a sequence of intervals $\Delta t_1, \Delta t_2, \Delta t_3, \ldots,$ where the sth interval is defined as $\Delta t_s = \{t | t_s' < t < t_s''\}$, t_s' and t_s'' are the beginning and the end of the interval, and $|\Delta t_s| = t_s'' - t_s'$ is the length of the interval. In some implementations of spiking neurons, the sequence of intervals might represent a continuous time set, i.e. $t_s'' = t_{s+1}'$, while in others there might be an explicit time shift between the separate intervals of synchronous firing, i.e. $t_s'' < t_{s+1}'$. For each interval we can define $\bar{t}_s$ as the mean time of the spikes in Δt_s. Examining the spikes from time $|\Delta t_s|$ before t_s and $|\Delta t_s|$ after t_s, that is in interval $2|\Delta t_s|$ around t_s, we can define a *spike time preference* of a neuron (threshold gate column) in the interval Δt_s as:

$$a_s^i = \begin{cases} 1 - \dfrac{|t_s^i - \bar{t}_s|}{\Delta t_s} & \\ \quad \text{if neuron (column) } i \text{ has fired in } 2|\Delta t_s| & \\ 0 & \\ \quad \text{if neuron (column) } i \text{ has not fired in } 2|\Delta t_s| & \end{cases}$$

Here, t_s^i denotes the firing time of neuron (column) i. Then the vector $a_s = (a_s^1, a_s^2, \ldots, a_s^N)$ is the *c-preference vector of cell assemblies of single neurons or threshold gate columns* in the time interval Δt_s. According to the above definition of a_s, a more synchronous firing in the assembly will lead to values in the preference vector close to 1. Alternatively, lower density of the spikes inside the time window will lead to values close to 0.5. Finally, firing times outside the time window will lead to values close to 0.

A *c-preference class of cell assemblies of single neurons or threshold gate columns* can be interpreted as a set of all preferences that represent the cell assembly for the same information with equal strength. This interpretation of the classes allows us to abstract from the particular distribution of the synchronous spikes in the time window usually considered as noise in biological systems.

3.1.2. Cortical column as a population of neurons

A second interpretation of the behavior of a single cortical column is when a column is considered to be a population of neurons representing one particular feature and the level of activation of that feature is determined by the relative number of excitatory neurons that have fired at a particular time, i.e. examining the population code of a single column. Such a concept is a computationally efficient approach for encoding features with analog values. It allows the combination of two different encoding schemes within a single network: graded activation of features as a population code of a single column and binding of features via synchrony firing of cell assemblies.

Let us now consider such a column i with P^i excitatory and Q^i inhibitory neurons. For a particular time interval Δt_s of synchronous firing, the number of excitatory neurons in column i that have fired is denoted as p_s^i, and the number of inhibitory neurons would be q_s^i respectively. We can define a value representing the activity of the column as:

$$a_s^i = \frac{1}{2}\left(1 + \frac{p_s^i}{P^i} - \frac{q_s^i}{Q^i}\right)$$

If most of the excitatory neurons in the column have fired and there is no activity of the inhibitory neurons, the activation value will be close to 1 and therefore indicate a strong preference for the feature that the column represents. The opposite situation will have a value close to 0 and would indicate strong suppression of the feature in the time interval. Finally, an activation value close to 0.5 would indicate low activity in the column and therefore no activation or suppression of the represented feature.

The vector constructed from the above defined activation values for all columns in a network with N columns $a_s = (a_s^1, a_s^2, \ldots, a_s^N)$ is the *c-preference of cell assemblies of cortical columns using population code*. A particular c-preference would represent the state of the network at a particular time and therefore contain a representation of the complex object (event) activated in the network at that time.

The *class of c-preferences of cell assemblies of cortical columns using population code* will allow us to abstract from the mutual fluctuations in the activity of the features included in a particular object. Such a class will include all c-preferences that represent the same object (event) with equal total activity of the assembly. Furthermore, the corner preference of the class will represent the object (event) as a binary vector and classes with the same corner preference will represent the same entity but with a different strength.

3.2. Dynamic representations: synfire chains and c-preferences

Sequential processing is essential for human language and reasoning. The neuroscience evidence suggests that there are dynamic structures in the brain that are essential and effective for sequential processing. After having introduced cell assemblies, we now look into some more dynamic representations, i.e. synfire chains. The concept of synfire chains as a model of cortical function was introduced by Abeles (1982, 1991). It explains some phenomena of precise timings in spatio-temporal patterns in frontal areas of the brain. A synfire chain consists of a precisely timed repeating sequence of synchronously firing small pools of neurons. The firing time in a chain can spread over a large time period – usually a few hundred milliseconds, or up to one second. The neural pools are linked together in a feed-forward chain, so that a wave of activity propagates from pool to pool in the chain. It has been shown that multiple spatio-temporal patterns can be stored in a network constructed of synfire chains where one neuron or cortical column can participate in several chains or several times in the same synfire chain (Herrman, Hertz, & Plügel-Bennet, 1995).

It has been suggested that the activity waves in the synfire chains represent an elementary cognitive event (Bienenstock, 1995). Synfire chains can be applied as storage elements of an associative memory, recognition and recall of spatio-temporal patterns and as a possible physical substrate for short term memory (Wennekers & Palm, 1999). Furthermore, it has been shown that synfire chains are able to regenerate ordered sequences of patterns (Aertsen, Diesmann, & Gewaltig, 1996; Abeles, Vaadia, Bergman, Prut, Headman, & Slovin, 1993; Bienenstock, 1995). Cell assemblies and synfire chains may also provide a way to explain elementary ways for structure processing in language using dynamic continuous stack mechanisms (Pulvermüller, 1999).

Synfire chains can be viewed as a possible extension of the associative memories from static spatial patterns to dynamic spatio-temporal ones. Furthermore, there are several properties of the synfire chains that result from their dynamic behavior. For example, the firing patterns exhibit cyclic activity, the order of firing of the pools in a chain is believed to be of significance, and different chains can share the same pool at different times without crosstalk (Hebb, 1949; Pulvermüller, 1999; Herrman, Hertz, & Plügel-Bennet, 1995). We suggest that a synfire chain is best interpreted as a dynamic symbolic representation and we propose the concept of preference machine as one possible solution.

3.2.1. C-preferences in the synfire chains

In an artificial neural network model, a synfire chain would represent a composite cognitive event. The event consists of several entities which might have explicitly defined semantics. Each pool in the chain would represent a single composite concept (entity). Therefore we can represent the activation of the network of N columns at a given interval Δt_s as a c-preference $a_s = (a_s^1, a_s^2, \ldots, a_s^N)$, where each value in the preference vector equals the population activity of a given column in the network. Such an interpretation is analogous to c-preferences of cell assemblies of a cortical column with population code and the formulas defined above are valid here:

$$a_s^i = \frac{1}{2}\left(1 + \frac{p_s^i}{P^i} - \frac{q_s^i}{Q^i}\right).$$

If the network has well-defined, say N, pools of columns, the c-preference can be constructed based on the population activity of these pools. In this case $P^i = Q^i$ is the total number of columns in pool i, and p_s^i denotes positive activity, i.e. is the number of excitatorily activated columns of pool i in interval Δt_s. Similarly, q_s^i denotes the negative activity in the pool, i.e. the number of inhibitorily activated columns.

3.2.2. Synfire chains as a preference machine

A c-preference will represent the activity of one or several pools in the chain at a particular time. To integrate the sequence of firing pools in the synfire chains, we develop a sequence of c-preferences representing the activity at each time step. Then we can construct a preference Moore machine that would be able to represent the behavior of synfire chains in the network. A direct interpretation of the representation of a cognitive event in the synfire chain (which is an ordered sequence of synchronous firing of neuronal pools in the chain) would be a final state (or set of states) of a preference Moore machine. If the network has activated only one event, the final state would be the one representing that event. Similarly, if the network activates several cognitive events, the preference machine will have multiple final states at the end, each representing a particular event. The intermediate state of the machine represents the history of the firing patterns of the network. A repeated intermediate state sequence indicates the cycling activity of the network. After this description of our general framework, we will turn to a concrete example in the next section.

4. Example: auditory network of spiking neurons with c-preference analysis

In this section we describe experiments with a model of pulsed neural network and preference analysis. These experiments are part of research to explore the properties of networks of spiking neurons and design models for processing complex temporal sequences.

The recognition of events represented with complex temporal sequences is a critical task performed in many perceptual and cognitive systems in the brain. It requires continuous combined spatial and temporal integration of information. Despite the extensive ongoing research, there is no general comprehensive understanding of how the biological systems represent and process information over time (Wermter, Austin, & Willshaw, 2001). Nevertheless, several encoding and processing schemes are known to be particularly useful for certain tasks and have been observed in the brain (Abott & Sejnowski, 1999).

There are some important indications on the type of information representation and processing for several sensory problems. There is a general agreement in neuroscience, and recently in the artificial neural networks communities, that the temporal dimension of the neural signals, i.e. the precise relative timing of the spikes, allows a powerful encoding and processing apparatus to be used, and is widely applied across the brain. The model presented in this section uses the relative timing of the onset, offset and peak events of the amplitude at different frequency channels of an input auditory signal (Jurafsky & Martin, 2000; Denham & Denham, 2001).

We describe an example experiment, where we have successfully applied the preference framework in order to extract and analyze the output of a network of spiking neurons. In order to study this initially, we developed a model of a c-preference analysis of clusters of neurons (cell assemblies) in a network of spiking neurons (Fig. 4). The task of the model is to recognize a brief complex sound. In particular, we focus on the task of recognizing auditory input, in this case of spoken words of the digits from one to nine.

4.1. Preprocessing of input data

The data consists of 10 single speaker's utterances for each digit, taken from the TIMIT database. As part of the preprocessing, the speech signal is filtered into 20 frequency channels on the Mel-scale spanning from 100 Hz to 4 kHz (Fig. 5). Furthermore, we consider the power and the times of onset, peak and offset of the amplitude in each channel. The generated input signal corresponding to each of the three events is described by the time at which this event has occurred in that frequency channel. The amplitude for the signal for the event e in the frequency channel f is given by the equation

$$A_{ef} = (1 - \sigma_a)\frac{\max(P_f)}{\max(P)} + \sigma_a$$

where $\max(P_f)$ and $\max(P)$ are the maximum power for the frequency channel and for the whole utterance, respectively. σ_a is a scaling factor that allows a controlled boost for those channels with little power. Furthermore, the signal for each event has a decay time τ_{ef} constant being proportional to its amplitude and a global time constant τ_l, i.e.

$$\tau_{ef} = \tau_l A_{ef}$$

Therefore, for a single utterance, the data extracted from the 20 frequency channels consists of 60 input signals describing the onset, peak and offset events in the channel. Each signal is associated with an amplitude and a time constant, and occurs at the time of the corresponding event in the frequency channel.

4.2. Pulsed neural network model

4.2.1. Neuron models and architecture

The details of pulsed neural networks in general (Maass & Bishop, 1999) are outside the scope of this paper, but we will briefly describe our architecture, neuron models and connectivity of our architecture. Then we will focus more on the preference analysis and experimental results.

The pulsed neural network is built in two layers. The details of the network presented here are the result of several experiments and tests with network and neuron parameters. The input layer contains 60 units – one unit per each event (onset, peak, offset) and 20 frequency channels (Fig. 6). At time t_{ef} when the corresponding event occurs for the first time in the channel, the unit starts generating a decaying continuous output current with an initial amplitude $I_{ef}(t_{ef}) = A_{ef}$ and time course given by the equation

$$\tau_{ef}\frac{dI_{ef}}{dt} = -I_{ef}(t).$$

An example of the input layer activity for an utterance of the word ''one'' is shown in Fig. 6.

The second layer of the network consists of fast spiking neurons modeled as leaky integrate-and-fire

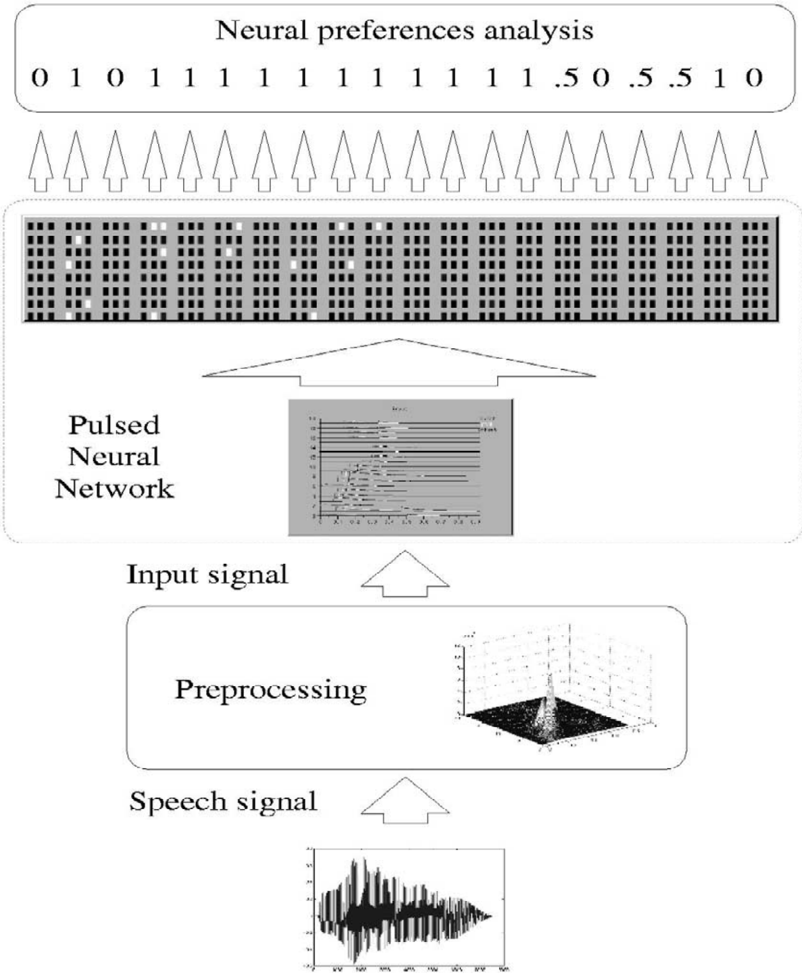

Fig. 4. The model of pulsed neural network and preference analysis for auditory signal recognition.

(lIAF) neurons (Maass & Bishop, 1999). The neurons are grouped in twenty clusters corresponding to each of the twenty frequency channels. Inside each cluster, the neurons are grouped in three columns – one for each type of event that is monitored in the channels (Fig. 7). There are 8 neurons in each column, making a total of 480 lIAF neurons in the second layer.

4.2.2. Connectivity

Units in the input layer feed directly into the body of the neurons in the second layer. Each input unit is connected with a probability of 0.8 to the 8 neurons in the corresponding cluster for frequency channel and event type. Each input unit is connected to at least one neuron at the second layer. This organization makes the neurons in a cluster primarily sensitive to events and activity in the corresponding frequency channel.

The neurons within a cluster in the second layer are connected to each other with probability 0.6 and the neurons in different clusters are connected with probability of 0.4. The synapses are allowed to change their sign, that is to become inhibitory from excitatory and vice versa. The weights are kept normalized with a total length of the weight vectors of 0.3 for the links inside the cluster and 0.2 for the links between different clusters. The connectivity

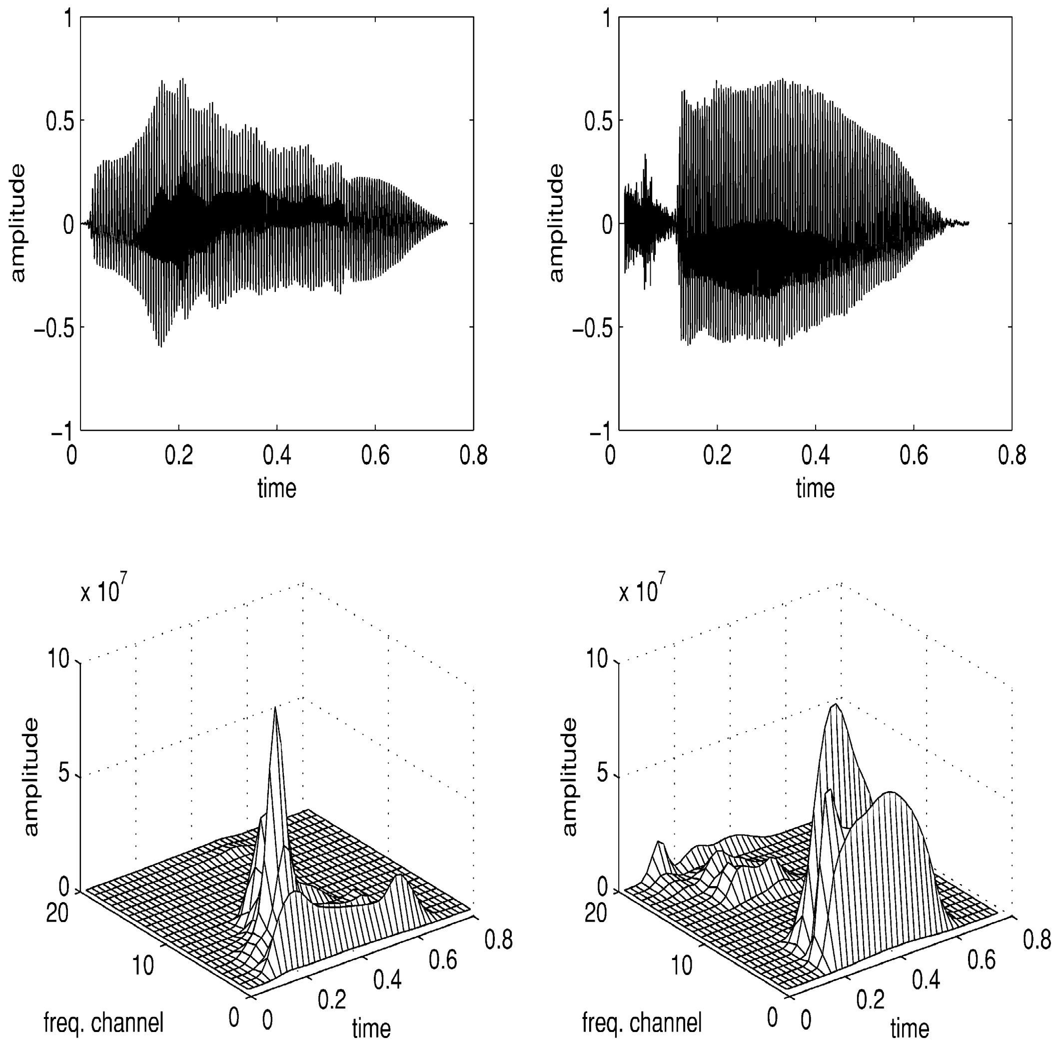

Fig. 5. Examples of the speech input (upper graphs) and the frequency diagrams (lower graphs) of utterances for the words ''one'' (left graphs) and ''two'' (right graphs) used in training.

probabilities and weight vectors lengths are empirically derived in order to optimize the network performance.

4.2.3. Processing

The network was trained using 5 utterances for each word. The auditory signal recognition takes part in the second layer of the network. Depending on the input signal, different clusters in the layer become active in parallel and enter a mode of decaying oscillation. Towards the end of the input signal and for a short period afterwards, the clusters form a weakly coupled oscillator with slowly decaying amplitude. Considered as a cell assembly where each cluster is an analog of a cortical column, the formation was observed to activate a set of features describing the recognized utterance (spoken word) as a response of the network to the input signal. For identifying the set of these features for each word

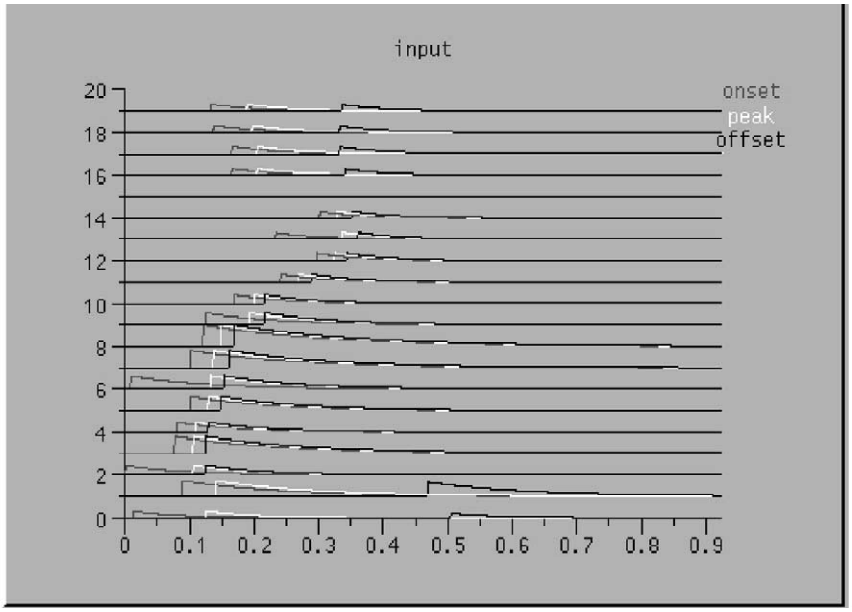

Fig. 6. Activation of the input neurons in processing the word ''one'' shown in Fig. 5 with $\sigma_a = 0.3$ and $\tau_l = 200$ ms. The horizontal axis shows time in seconds. The 20 frequency channels are superimposed with step 1 at the ordinate. At each step, there are three graphs representing the activation level of each of the three neurons in that channel.

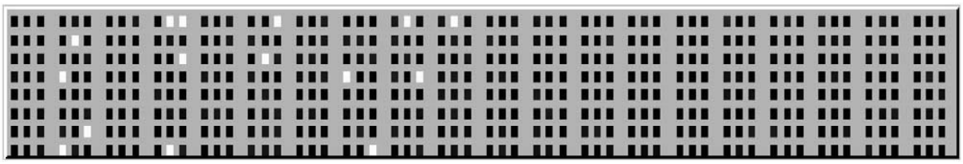

Fig. 7. Firing neurons (white squares) in the second layer at the end of the input signal of the utterance of the word ''one'' shown in Figs. 5 and 6 (that is 617 ms). Lighter colors indicate higher potential and the white color indicates that the neuron has just emitted a spike.

and reading the network's response, we used complex preferences.

4.3. The preference analysis

For the purpose of this experiment we have adapted the general definition of c-preference for cell assemblies of cortical columns as population code (see Section 3.1.2) into a more specific one taking into account the properties of the data and the organization and dynamics of the neurons in our model. We also use features of c-preferences of cell assemblies of cortical columns as threshold gates.

We consider each cluster of neurons in the second layer to be an analog of a cortical column, or more precisely a population of columns which behaves as a single functional unit. Therefore, our model has 20 ''cortical columns'' participating in different cell assemblies. Most of the columns participate in several assemblies and in the experiment reported here some columns are found not to be reliably active in any assembly.

The clusters in the model fire with a dynamic (decaying) mean firing rate. This effectively means that a simple time interval Δt which could be used in measuring the activity of the cluster/column over a representative number of intervals could not be found. Δt either has to be: (1) a dynamic interval possibly related to the decay rate or the total activity of the network; or (2) the activity of a column has to be averaged over a representative period of time. In our approach, the activity of the columns was examined for a fixed period of time around the end of the input signal.

Furthermore, in our model, there is no distinction between excitatory and inhibitory neurons in the column. As expected after learning, an examination of the links within a column and between the columns showed that there is an overall excitation between the neurons in the same column and that inhibition was primarily coming from neurons in other columns. In other words, a high number of firing neurons in the column indicates higher total excitation of that column. Therefore a good approximate value for the activation of the column would be one that is proportional to the mean firing rate of the population of neurons in it.

Based on the above observations, we derived the following equation for the activity a^i of column i in our model

$$a^i = \begin{cases} \dfrac{1}{2}\left(1 + \dfrac{p^i}{P}\right) & \text{if } p^i > \Delta t \\ \dfrac{p^i}{2P} & \text{if } p^i \le \Delta t \end{cases}$$

where p^i is the number of spikes from column i in the period Δt and P is the maximum number of spikes observed in any column for that period. The value of activity of a column would be greater than 0.5 if the column has had on average at least one spike per millisecond during the observation period, and less than 0.5 otherwise. Normalization with the maximum number of spikes found in that period allows us to abstract from fluctuations in the absolute power in the input speech signal and concentrate on the relative activity of the columns.

The c-preference $(a^1, a^2, \ldots, a^{20})$ will represent the cell assembly that has been active at the end of the input speech signal. The next corner reference would be a matching pattern for a symbolic representation of the recognized word.

Based on the recorded activity of the cluster in the network while processing the 45 training patterns (5 training utterances for each word) and the analysis of the network weights, we derived partial next corner references for each word (Table 1). These references are automatically derived from the trained network output. This process is general and does not depend on the particular network. A value of 0 in the partial next corner reference indicates that the column does not respond to speech signals of that word, 1 indicates a strong response, and $\frac{1}{2}$ indicates that the column has no reliable negative or positive response to an input for the word. For example, the matching reference for the word "one" shows that on the training examples, the neurons responding to events in the 1st and 3rd frequency channel have low activity and therefore these neurons will be expected to have a low response when any test speech signal of the word "one" is presented. In contrast, the

Table 1
Sample patterns for the words as partial next corner reference

	Partial next corner reference in the 20 frequency channels																			
Word	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
"one"	0	1	0	1	1	1	1	1	1	1	1	1	1	1	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	1	0
"two"	0	1	1	1	1	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	0	0	0
"three"	0	1	$\frac{1}{2}$	1	1	0	0	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
"four"	0	1	0	1	1	1	1	1	$\frac{1}{2}$	1	1	1	1	1	$\frac{1}{2}$	0	0	0	0	0
"five"	0	1	0	1	0	1	1	1	1	1	1	1	1	1	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0
"six"	0	1	0	1	1	1	1	$\frac{1}{2}$	$\frac{1}{2}$	0	0	$\frac{1}{2}$	0	0	0	1	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	0
"seven"	0	1	0	1	1	1	1	1	1	1	1	1	1	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
"eight"	0	1	1	1	1	1	1	$\frac{1}{2}$	0	0	0	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
"nine"	0	1	0	1	1	1	$\frac{1}{2}$	1	1	1	1	1	$\frac{1}{2}$	0	0	0	0	$\frac{1}{2}$	$\frac{1}{2}$	0

neurons responding to events in the 2nd, 4th, etc. channels show high activity at the end of the training examples of ‘‘one’’. Finally, the neurons in the 15th, 17th and 18th columns do not show reliable (high or low) activity in response to ‘‘one’’ and therefore are not examined when processing test examples. Effectively, we use the c-preference analysis of the behaviour of the network on the training examples in order to build a filter of the network output.

4.4. Testing and results

The performance of the model has been tested with the remaining 45 patterns (5 untrained utterances for each word). The spikes for each column in the second layer of the network are being counted for 200 ms, starting at 100 ms before the end of the input signal. It may be argued that the three times at which we measure the signal are restricted. However, this is the period where the activity of the cluster is most representative. We examine the output of the network, not the input. There is a delay between the events in the input pattern and the response of the cluster. Therefore, although the most significant events in the input signal are observed just after onset and just before offset, the significant activity in the output cluster is towards the end of the signal and just after that. The output c-preference and the next corner reference of the network's response are computed based on spike numbers this. The resulting output reference is tested for matching the sample patterns shown in Table 1 using the following condition: if the value of the sample pattern is not $\frac{1}{2}$ the value in the output reference has to be the same as the corresponding value in the sample pattern. If the value in the sample pattern is $\frac{1}{2}$, i.e. the corresponding column is not significant for the recognition of that word, the value of the output reference for that column is not examined.

Table 2 shows the results obtained for the training and test sets. We used a standard evaluation metric of recall and precision rates (Salton, 1989) for evaluating the performance. The performance of the models depends mainly on the individual performance of the neural network and the c-preference analysis. The good final results demonstrate the ability of the pulsed neural networks and that the analysis with c-preferences is an adequate interpretation and is able to capture the output behaviour of a neural network of spiking neurons. For most of the digits, we achieved recall of 0.8 or higher, which means that the model was able to recognize at least 80% of the test patterns. The precision for many of the examples is 1 which indicates that almost all classifications made by the model are correct.

Table 2
Recall and precision results from the recognition of auditory signals with our model

	Training set		Test set	
Word	recall	precision	recall	precision
‘‘one’’	0.8	1	1	1
‘‘two’’	0.6	1	1	1
‘‘three’’	0.6	1	1	0.83
‘‘four’’	0.8	1	0.6	0.6
‘‘five’’	0.8	1	0.8	1
‘‘six’’	0.8	1	0.6	0.75
‘‘seven’’	1	0.83	0.4	1
‘‘eight’’	1	1	1	1
‘‘nine’’	0.8	0.67	0.8	0.67
Total	0.8	0.95	0.8	0.86

5. Discussion and conclusion

Architectural abstractions at different levels are important in order to link higher level cognitive functions like language processing with the neuroscience evidence from the brain. The complexity of cognitive and neurobiological processes makes it seem plausible that several representational levels may be advantageous (Gutknecht, 1992; Sun, 1996). In this context we have explored hybrid symbolic/connectionist machines and their relationship to neuroscience evidence. In particular we have explored the use of preference machines as one particular type of hybrid sequential machines. Then we have introduced c-preferences which allow the integration of various hybrid encoding schemes like mean firing rate, time-to-first spike or synchrony encoding and information representation structures like cell assemblies and synfire chains. The introduction of complex preferences was motivated by these complementary encoding schemes. We argue that it is necessary to integrate various neural encoding schemes of underlying neuronal processing and that symbolic, connectionist and neuroscience levels can be integrated using the preference framework. By considering the neuroscience level, important new insights can be gained for higher symbolic connectionist levels.

There has been a lot of work on hybrid symbolic/connectionist architectures in the last decade (Wermter & Sun, 2000; Medsker, 1994a; Goonatilake & Khebbal, 1995; Medsker, 1994b; Hilario, Lallement, & Alexandre, 1995; Sperduti, 1994; Rocha & Yager, 1992; Sun, 1994; Sun & Bookman, 1995). Furthermore, there has been early, but very different symbolic work on preference semantics as a means for high level structural frames and concepts (Wilks, 1978; Wilks, Huang, & Fass, 1985). However, neuroscience concepts like cell assemblies and synfire chains have not yet been integrated into cognitive hybrid architectures in the past. The main contribution of this paper is to create a first link and to demonstrate how concepts from hybrid architectures, in particular preference Moore machines, can be linked to novel concepts from neuroscience. Furthermore, the preference framework has integrated several neural encoding schemes.

The main focus of this paper is to provide theoretical background to hybrid sequential machines based on inspiration from neuroscience. This fundamental view has been taken since it has been argued that hybrid architectures of the past have been too task- and domain-specific and since more general notions towards a theory of hybrid machines have to be developed (Wermter & Sun, 2000; Medsker, 1994b). However, in one concrete example we have demonstrated how these concepts can be applied to a pulsed neural network for auditory processing. There has been very little work on learning in pulsed neural networks so far and therefore the development of such a network within the preference framework demonstrates that such complex dynamic neural architectures can process auditory input successfully. Of course, from a speech application point of view, digit recognition has been realized many times before. However, this is the first time a pulsed neural network based on dynamic spiking processing has been trained successfully for this task. Furthermore, we have demonstrated how this neuroscience-inspired network can be linked to the preference framework.

It is envisaged that tasks like auditory processing, robust syntactic analysis or semantic classification could benefit especially from such approach. Previously, we have demonstrated how recurrent connectionist preference machines have been used for classifying short text titles (Wermter, Panchev, & Arevian, 1999; Wermter, Arevian, & Panchev, 1999a) or shallow spoken language parsing (Wermter & Weber, 1997). This work demonstrated the preference framework at the connectionist and symbolic levels (Wermter, 2000a). Furthermore, there is new challenging work underway, which further extends the link from neuroscience-inspired level to higher preferences. For instance, we have integrated various neural encoding schemes using preferences (Panchev & Wermter, 2000) and have explored a neuroscience-inspired mirror neuron system for analysing finite state sequences (Womble & Wermter, 2001).

For future work, we plan to examine cell assemblies and synfire chains further in the preference framework. As we demonstrated, cell assemblies can be interpreted and linked to preferences. Furthermore, synfire chains can be linked to preference Moore machines. So far, existing dynamic neuroscience-inspired architectures have the ability to use temporal processing but learning and interpretation of their knowledge in a connectionist or symbolic form is needed for two reasons: First, from a neurocognitive point of view, knowledge in the brain can be communicated and the integration of high level cognition and brain level architectures is necessary. Second, from a representational and computational point of view, the link of dynamic neuroscience networks with connectionist or symbolic machines allows to take advantage of existing hybrid sequential machines.

Acknowledgements

This research has been partially supported by EPSRC grant GR/M56555. We would like to thank Garen Arevian and Mark Elshaw for their comments on this paper.

References

Abeles, M. (1982). *Local cortical circuits: an electrophysiological study*. Springer-Verlag.

Abeles, M. (1991). *Corticonics: neural circuits of the cerebral cortex*. Cambridge University Press.

Abeles, A., Vaadia, E., Bergman, H., Prut, Y., Headman, I., & Slovin, H. (1993). Dynamics of neuronal interactions in the frontal cortex of behaving monkeys. *Concepts in Neuroscience*, *4*, 131–158.

Abbott, L., & Sejnowski, T. (Eds.), (1999). *Neural codes and distributed representations: foundations of neural computation*. MIT Press.

Aertsen, A., Diesmann, M., & Gewaltig, M. O. (1996). Propagation of synchronous spiking activity in feedforward neural networks. *Journal of Physiology (Paris)*, *90*, 243–247.

Anderson, J. A., & Rosenfeld, E. (Eds.), (1988). *Neurocomputing: foundations of research*. Cambridge: MIT Press.

Bienenstock, E. (1995). A model of neocortex. *Network*, *6*, 179–224.

Denham, S., & Denham, M. (1999). An investigation into the role of cortical synaptic depression in auditory processing. In *Proceedings of the EmerNet: international workshop on emergent neural computational architectures based on neuroscience, ICANN '99*, Edinburgh, Scotland.

Denham, S., & Denham, M. (2001). An investigation into the role of cortical synaptic depression in auditory processing. In S. Wermter, J. Austin & D. Willshaw (Eds.), *Emergent neural computational architectures based on neuroscience (this volume).* Heidelberg, Germany: Springer-Verlag.

Elman, J. L., Bates, E. A., Johnson, M. H., Karmiloff-Smith, A., Parisi, D., & Plunkett, K. (1996). *Rethinking innateness*. Cambridge, MA: MIT Press.

Feldman, J. A., & Ballard, D. H. (1982). Connectionist models and their properties. *Cognitive Science*, *6*, 205–254.

Fransén, E., & Lansner, A. (1998). A model of cortical associative memory based on a horizontal network of connected columns. *Network*, *9*(2), 235–264.

Fransén, E., Lansner, A., & Liljenström, H. (September 9–10 1994). A model of cortical associative memory based on Hebbian cell assemblies. In L. F. Niklasson & M. B. Bodén (Eds.), *Connectionism in a broad perspective: selected papers from the Swedish conference on connectionism – 1992.* Skövde, Sweden: Ellis Horwood, pp. 165–172.

Goonatilake, S., & Khebbal, S. (1995). *Intelligent hybrid systems*. Chichester: Wiley.

Gutknecht, M. (1992). The postmodern mind: hybrid models of cognition. *Connection Science*, *4*, 339–364.

Hebb, D. O. (1949). *The organization of behavior*. New York: Wiley, Partially reprinted in (Anderson & Rosenfeld, 1988).

Herrman, M., Hertz, J., & Plügel-Bennet, A. (1995). Analysis of synfire chains. *Network: Computation in Neural Systems*, *6*(3), 403–414.

Hilario, M., Lallement, Y., & Alexandre, F. (1995). Neurosymbolic integration: unified versus hybrid approaches. In *Proceedings of the European symposium on artificial neural networks*, Brussels.

Hopcroft, J., & Ullman, J. (1979). *Introduction to automata theory, languages, and computation*. Reading, MA: Addison Wesley.

Jurafsky, D., & Martin, J. (2000). *Speech and language processing*. Prentice Hall, Reviewed.

Maass, W., & Bishop, C. (1999). *Pulsed neural networks*. MIT Press.

Manolios, P., & Fanelli, R. (1994). First-order recurrent neural networks and deterministic finite state automata. *Neural Computation*, *6*, 1155–1173.

the PDP Research Group, McClelland, J. L., & Rumelhart, D. E. (1986). Parallel distributed processing: explorations in the microstructure of cognition, vol. 2. Cambridge, MA: MIT Press.

Medsker, L. R. (1994a). *Hybrid neural network and expert systems*. Boston, MA: Kluwer Academic Publishers.

Medsker, L. R. (1994b). *Hybrid neural network and expert systems*. Boston: Kluwer Academic Publishers.

Omlin, C. W., & Giles, C. L. (1994). Extraction and insertion of symbolic information in recurrent neural networks. In V. Honavar & L. Uhr (Eds.), *Artificial intelligence and neural networks: steps towards principled integration*. San Diego: Academic Press, pp. 271–299.

Omlin, C. W., & Giles, C. L. (1996a). Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, *9*(1), 41–52.

Omlin, C. W., & Giles, C. L. (1996b). Stable encoding of large finite-state automata in recurrent neural networks with sigmoid discriminants. *Neural Computation*, *8*(4), 675–696.

Palm, G. (1986). Associative networks and cell assemblies. In G. Palm & A. Aertsen (Eds.), *Brain theory*. Springer.

Palm, G. (1993). On the internal structure of cell assemblies. In A. Aertsen (Ed.), *Brain theory*. Amsterdam: Elsevier, pp. 261–270.

Panchev, C., & Wermter, S. (2000). Complex preferences for the integration of neural codes. In *Proceedings of the international joint conference on neural networks*, Como, Italy, pp. II-253–II-258.

Pulvermüller, F. (1999). Words in the brain's language. *Behavioral and Brain Sciences*, *22*, 253–336.

Rocha, A. F., & Yager, R. (1992). Neural nets and fuzzy logic. In Kandel, A., & Langholz, G. (Eds.), *Hybrid architectures for intelligent systems*. Boca Raton: CRC Press Inc, pp. 4–28.

Rumelhart, D. E., Hinton, G. E., Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing*, vol. 1. Cambridge, MA: MIT Press, pp. 318–362.

Salton, G. (1989). *Automatic text processing*. New York: Addison-Wesley.

Shastri, L. (2001). Biological grounding of recruitment learning and vicinal algorithms in long-term potentiation. In S. Wermter, J. Austin & D. Willshaw (Eds.), *Emergent neural computational architectures based on neuroscience (this volume).* Heidelberg, Germany: Springer-Verlag.

Shavlik, J. (1994). A framework for combining symbolic and neural learning. In V. Honavar & L. Uhr (Eds.), *Artificial intelligence and neural networks: steps towards principled integration*. San Diego: Academic Press, pp. 561–580.

Sperduti, A. (1994). Labelling recursive auto-associative memory. *Connection Science*, *6*, 429–459.

Sun, R. (1994). Logics and variables in connectionist models: a brief overview. In V. Honavar & L. Uhr (Eds.), *Artificial intelligence and neural networks: steps towards principled integration*. San Diego: Academic Press, pp. 301–320.

Sun, R. (1996). Hybrid connectionist-symbolic models: A report from the IJCAI95 workshop on connectionist-symbolic integration. *Artificial Intelligence Magazine*.

Sun, R., & Bookman, L. A. (1995). *Computational architectures integrating neural and symbolic processes*. Boston, MA: Kluwer Academic Publishers.

Taylor, J. (1999). Images of the mind: brain imaging and neural networks. In S. Wermter, J. Austin & D. Willshaw (Eds.), *Proceedings of the EmerNet: international workshop on emergent neural computational architectures based on neuroscience*, ICANN '99, Edinburgh, Scotland.

Thorpe, S., Fize, D., & Marlot, C. (1996). Speed in the processing in the human visual system. *Nature*, *384*, 520–522.

Towell, G. G., & Shavlik, J. W. (1994). Knowledge-based artificial neural networks. *Artificial Intelligence*, *70*(1/2), 119–165.

Wennekers, Th. & Palm. G. (1999). Cell assemblies, associative memory and temporal structure in brain signals. In R. Miller (Ed.), *Time and brain. Conceptual advances in brain research*, vol. 2. Harwood Academic Publishers, to appear.

Wermter, S. (1999). Preference Moore machines for neural fuzzy integration. In *Proceedings of the international joint conference on artificial intelligence*, Stockholm, pp. 840–845.

Wermter, S., Arevian, G., & Panchev, C. (1999a). Recurrent neural network learning for text routing. In *Proceedings of the international conference on artificial neural networks*, Edinburgh, UK, pp. 898–903.

Wermter, S., Austin, J., & Willshaw. D. (1999). *Proceedings of the Emernet international workshop on emergent neural computational architectures based on neuroscience*. http://www.his.sunderland.ac.uk/emernet/Publications_frame.html, Edinburgh.

Wermter, S., Austin, J., & Willshaw, D. (Eds.), (2001). *Emergent neural computational architectures based on neuroscience*. Springer.

Wermter, S., Panchev, C., & Arevian, G. (1999). Hybrid neural plausibility networks for news agents. In *Proceedings of the national conference on artificial intelligence*, Orlando, USA, pp. 93–98.

Wermter, S., & Sun, R. (2000). *Hybrid neural systems*. Heidelberg: Springer.

Wermter, S. (2000a). Knowledge extraction from transducer neural networks. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solvin.*, *12*, 27–42.

Wermter, S. (2000b). Neural fuzzy preference integration using neural preference moore machines. *International Journal of Neural Systems*, *10*(4), 287–309.

Wermter, S., & Weber, V. (1997). SCREEN: Learning a flat syntactic and semantic spoken language analysis using artificial neural networks. *Journal of Artificial Intelligence Research*, *6*(1), 35–85.

Wilks, Y. (1978). Making preferences more active. *Artificial Intelligence, 11.*

Wilks, Y., Huang, X., & Fass, D. (1985). Syntax, preference and right attachment. In *Proceedings of the ninth international joint conference on artificial intelligence*, Los Angeles, CA, pp. 779–784.

Womble, S., & Wermter, S. (2001). A mirror neuron system for syntax acquisition. In *International conference on artificial neural networks*.