ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

Modeling Efficient Conjunction Detection with Spiking Neural Networks

Sander M. Bohte[1], Joost N. Kok[1,2] and Han La Poutré[1,3]
[1]CWI, Kruislaan 413, 1090 GB Amsterdam, P.O. Box 94079, The Netherlands
[2]LIACS, Leiden University, 2300 RA Leiden, P.O. Box 9512, The Netherlands
[3]School of Techn. Manag., TU Eindhoven, The Netherlands

Abstract. The design of neural networks that are able to efficiently encode and detect conjunctions of features is an important open challenge that is also referred to as "the binding-problem". We define a formal framework for neural nodes that process activity in the form of tuples of spike-trains which can efficiently encode and detect feature-conjunctions on a retinal input field in a position-invariant manner, also in the presence of multiple feature-conjunctions.

1 Introduction

The representation of structured information in neural networks has so far not been satisfactory solved, though it is thought to be required for efficiently solving a number of notoriously hard problems [1]. In a linguistic sentence like *The red apple and the green pear*, grammar implies the structuring of elements "red", "green", "apple", and "pear" into semantic composites, in brackets: {{*red,apple*}, {*green,pear*}}. The *binding-problem* refers to the problem of how to encode and detect such structured representations in neural networks. Whereas we can identify elements like *red*, *green*, *apple*, and *pear* each with a corresponding (active) neuron, the embodiment of the structural brackets has been much debated, as far back as Hebb, in 1949. Some have even argued that structural representation is impossible in neural networks [2].

In the context of visual perception, the main concern is how to represent and/or detect conjunctions such as *red* and *apple*. Creating a *red apple* detector for every location on the retina seems too expensive, at least for every sensible conjunction. The straight-forward solution, as also depicted in figure 1A, would seem to first create position-invariant, or "global" *apple* and *red* detectors by combining the responses of the respective local detectors, and then detect conjunctions from these global detectors. However, this architecture is ambiguous and prone to errors in the presence of multiple conjunctions, because there are no structuring "brackets" in the neural activation encoding (e.g. [1, 3]): the link between *red* and *apple* and the link between *green* and *pear* is lost.

Recently, progress has been reported on structured representation in neuro-symbolic AI [4]. This solution uses *vectors* of binary neural activity as the primary data-structure, and binding is achieved via manipulation of these vectors to signify structure. To transfer these results to the field of computer-vision, and address the classical binding-problem, we need to solve the specific problem of global detection. Additionally, the solution should work in a feed-forward neural

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

network, as the (human) detection of whole objects (e.g. *red apple*) is a very fast process [5], suggesting that the combination of local features into wholes, e.g. {*red*},{*apple*} → {*red apple*}, is essentially feed-forward.

Here, we present the formal definition of a framework that addresses these problems. The framework enables position-invariant conjunction detection in a feed-forward neural architecture that processes tuples of spike-trains in its neural nodes. We remark that the framework can easily be implemented in spiking neural networks, but *cannot* in traditional networks of sigmoidal neurons.

2 Architecture

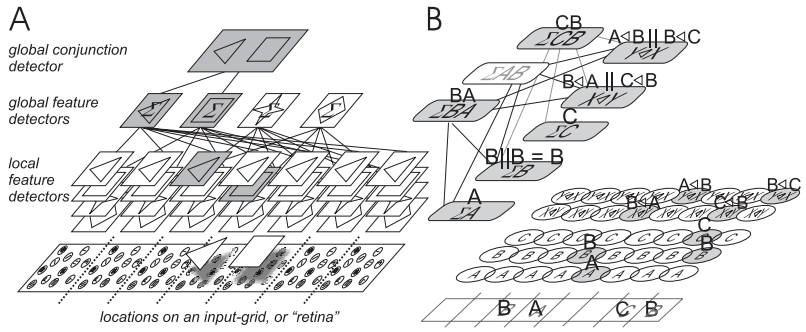

Figure 1: A) Global conjunction detection via local and global feature detectors. A square on the grid contains the same set of basic neurons. B) Tuple propagation when detecting feature-conjunctions.

The local detection of features such as a *square* can be considered in terms of tuples of activity. We assume that all (discrete) locations on an input grid are populated with identical sets of diversely tuned basic neurons (e.g. "retina" in fig. 1A). The presence of a feature like *square* is then characterized by the activity (spike-trains) it elicits in a set of basic neurons. The timings of the spike-trains are collected for each set in a tuple, where each tuple-element describes the spike-train of one neuron. The proposed architecture processes such tuples.

The architecture is depicted in figure 1B and C, where the local "nodes" process tuples of spike-trains (TST's). We have local feature-detecting nodes that look for a specific feature. Each such node looks at one set of basic neurons. We also have local nodes that detect and signal the conjunction of two arbitrary features. These nodes consider two sets of basic neurons. The idea is that they detect the presence of heightened activity in both locations, without considering what this represents in terms of actual features. In our example, these nodes look at two locations next to each other in the grid.

The next level in the architecture combines the results of local nodes. Here, we exploit a specific property of spiking neurons: global aggregate TST's can be obtained by combining the local TST's. Suppose we combine two n-tuples of spike trains in which in each of them there are in k different spikes, then the combined n-tuple contains $2k$ spikes. This property of spiking neurons alleviates the "superposition problem" that occurs in the case of *sigmoidal* neurons [1].

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

The local feature detecting nodes are combined in global feature detecting nodes ("there is a triangle in one of the locations") and global conjunction-presence detectors ("there are two active locations next to each other"). Then, finally, these global detectors are then combined to detect the presence of specific features next to each other (a feature-conjunction).

The detection of specific features-next-to-each-other from the global detectors is possible, because we make use of a special local procedure: the output of local next-to-each-other detectors is one of the two features, "*watermarked*" with the other feature, that is: some spike trains are shortened or even removed due to the presence of the other feature. Implicitly, the two features are then linked. The formal details of the architecture are given in the next section.

3 Formal Description

In the remainder, we use the notation like $(d, e, \ldots \in)D$ to introduce a domain D and some typical elements d, e. That is, for the subsequent use of a variable d or e, one can assume that it is an element of D.

DEFINITION 1 *Let $(a, b, \ldots \in)\mathbb{R}$ be the domain of single spikes.*
Let $(s, t, u, v, \ldots \in)ST \subseteq \mathbb{R}^*$ be the domain of spike-trains with as elements the strictly monotonically increasing sequences denoting the timing of single spikes. An empty sequence is denoted by ε. The concatenation operator is denoted by $\odot$: given single spike a and spike-train s, concatenation yields the new spike-train $a \odot s$, which is the same as s, except that the element a is added in front.

The set TST of tuples of n spike-trains is defined by: $(S, T, U, V, \ldots \in) TST = \prod_{i=1}^{n} \mathbb{R}^*$. By S_i we denote the projection of S on the i-th component. An empty TST is denoted by $\mathfrak{E}$ (i.e. $\mathfrak{E} = (\varepsilon, \ldots, \varepsilon)$).

An example of the use of the concatenation operator: $1 \odot < 2.6, 4 > = < 1, 2.6, 4 >$. A neuron may produce a spike-train $s = < 3.1, 7, 12.34 >$ or an empty spike-train ε. Some n neurons together produce a tuple S of spike-trains, e.g. $(S_1, S_2, \ldots, S_n)$.

DEFINITION 2 *The combination of two TST's is defined as $S \parallel T = (S_1 \parallel T_1, \ldots, S_n \parallel T_n)$, with $\varepsilon \parallel s = s$, $s \parallel \varepsilon = s$, and with s and t non-empty:*

$$(a \odot s) \parallel (b \odot t) = \begin{cases} a \odot (s \parallel (b \odot t)) & \text{if } a < b \\ b \odot ((a \odot s) \parallel t) & \text{if } b < a \\ a \odot (s \parallel t) & \text{if } a = b. \end{cases}$$

The operator Σ extends $\parallel$ and denotes the combination of more than two TST's.

EXAMPLE *The combination $S \parallel T$ of two TST's $S = (< 2, 3.1, 4 >, < 2.5, 3.1 >)$ and $T = (< 3.9, 4.2 >, \varepsilon)$ equals $(< 2, 3.1, 3.9, 4, 4.2 >, < 2.5, 3.1 >)$.*

DEFINITION 3 *For spike-trains s, t, we have $s \subseteq t$ if there is a spike train u such that $s \parallel u = t$. We say $S \subseteq T$ if for all projections $S_i \subseteq T_i$, $i = 1, \ldots, n$.*

EXAMPLE *For example, $< 1, 2, 3 > \subseteq < 0.5, 1, 1.3, 2, 3 >$ because $< 1, 2, 3 > \parallel < 0.5, 1.3 > = < 0.5, 1, 1.3, 2, 3 >$.*

Now, we define the *watermarking*-operator (Context-Dependent Thinning, CDT).

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

DEFINITION 4 *A CDT operation $\lhd$ for binding is defined by: $S \lhd T = (S_1 \lhd T_1, \ldots, S_n \lhd T_n)$, with: $t \lhd \varepsilon = t$, $\varepsilon \lhd t = \varepsilon$, and for non-empty spike-trains:*

$$(a \odot s) \lhd (b \odot t) \begin{cases} a \odot (s \lhd (b \odot t)) & \text{if } a < b \\ \varepsilon & \text{otherwise.} \end{cases}$$

Note that for all i, either $(S \lhd T)_i = \varepsilon$ or $(T \lhd S)_i = \varepsilon$ (or both equal ε). Another interesting property is that for all S, T, U, we have $S \lhd (T \parallel U) = (S \lhd T) \lhd U$.

EXAMPLE *The CDT operation on spike-trains $s =< 1, 2, 3 >$ and $t =< 2.5, 3.5, 4.6 >$, results in a spike-train $s \lhd t =< 1, 2 >$. The operation $S \lhd T$ removes some spikes in S due to the presence of spikes in T, that is $(S \lhd T) \subseteq S$.*

DEFINITION 5 *We define $\Gamma^m(S) = T$, with $T_i = \sum_{k=i-m}^{i+m} \begin{cases} S_n & \text{if } k \bmod n = 0 \\ S_{k \bmod n} & \text{otherwise.} \end{cases}$*

We abbreviate $S \lhd (\Gamma^m(T))$ by $S \lhd^m T$. In this abstract, we take $m = 0$, hence $S \lhd^m T = S \lhd T$. For cases with $m > 0$, see the full paper [6].

For the construction of networks, we next define local feature-detectors, local binding-detectors and conjunction-detectors.

DEFINITION 6 *A local feature-detector lfd for feature S in T is defined by:*

$$lfd(S, T) = \begin{cases} T & \text{if } S \subseteq T \\ \mathfrak{E} & \text{otherwise.} \end{cases}$$

DEFINITION 7 *A generic local binding-operator glb is defined by:*

$$glb(S, T) = \begin{cases} S \lhd^m T & \text{if } |S| \cdot |T| > \theta \\ \mathfrak{E} & \text{otherwise.} \end{cases}$$

Here, the number of spikes $|S|$ in S is defined by $|S| = |S_1| + \ldots + |S_n|$, with $|s| = \text{length}(s)$ and $|\varepsilon| = 0$. The threshold θ is the required input-activation.

The idea is that a non-empty TST, S, is partially propagated by the *glb* operator if there is also a non-empty TST, T, present (a conjunction). The CDT operation is then performed on S using T, resulting in a TST like S, except for that some elements from the spike-trains in S are removed. We use two complementary *glb* operators, denoted $(X \lhd Y)$ and $(Y \lhd X)$. When presented with *any* non-empty TSTs S and T, these operators yield watermarked versions of S or T, respectively. For global conjunction detection, the presence of the correct watermarked TST is determined by the Ω operator:

DEFINITION 8 *We define the presence operator Ω by:* $\Omega(S, T) = (\Omega(S_1, T_1), \ldots, \Omega(S_n, T_n))$, *where* $\Omega(s, t) = \begin{cases} s & \text{if } s \subseteq t \\ \varepsilon & \text{otherwise.} \end{cases}$.

The $\Omega(S, T)$ operation checks which spike trains of S are present in T and outputs those spike-trains that are present.

DEFINITION 9 *The conjunction detector $cd(S, T, U, V)$ for the S, T conjunction is defined by:*

$$cd(S, T, U, V) = \begin{cases} \Omega((S \lhd^m T), U) \parallel \Omega((T \lhd^m S), V) & \text{if } |\Omega((S \lhd^m T), U)| + \\ & |\Omega((T \lhd^m S), V)| \geq \alpha \\ \mathfrak{E} & \text{otherwise.} \end{cases}$$

The threshold α detects matching, we set it to to $|S \lhd^m T| + |T \lhd^m S|$.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

The conjunction detector propagates a specific mix of the input TST's if both sufficiently match the patterns. It checks whether watermarked versions of S and T are present in U and V, respectively.

Now we have all the (feed-forward) elements for our three-level architecture:

1. a first level in which we have:
   – for each location i local feature detectors $lfd(A, U^i)$, $lfd(B, U^i)$ looking for specific patterns $A, B \in$ TST in the activity U^i of the local set of neurons.
   – for pairs of locations next to each other generic local binding operators $glb(U^i, U^{i+1})$ and $glb(U^{i+1}, U^i)$ that look for pairs of sufficient activation, and then output watermarked features $U^i \lhd^m U^{i+1}$ and $U^{i+1} \lhd^m U^i$.
2. a second level combining local feature and generic binding detectors via $\|$ into respective global feature and generic binding detectors: $\Sigma_i lfd(A, U^i)$, $\Sigma_i lfd(B, U^i)$, $\Sigma_i glb(U^i, U^{i+1})$, $\Sigma_i glb(U^{i+1}, U^i)$.
3. a third level, for conjunction detection. A *cd*-operator connects to two global feature detectors and two global generic binding detectors: $cd(\Sigma_i lfd(A, U^i), \Sigma_i lfd(B, U^i), \Sigma_i glb(U^i, U^{i+1}), \Sigma_i glb(U^{i+1}, U^i))$.

Up to now we have abstracted from the fact that computations take time. In order to detect a feature a node can only produce output if it has seen the feature. Therefore, to actually implement the network, one has to build in a delay in all nodes in the network. This can be done by using a constant Δ, which models the maximal computation time.

DEFINITION 10 *For a number $\Delta \in \mathbb{R}$ the operation of delayed propagation Δ of a TST S, is defined as: $\Delta(S) = (\Delta(S_1), \ldots, \Delta(S_n))$, with the propagation of an empty sequence ε defined as $\Delta(\varepsilon) = \varepsilon$, and the propagation on a non-empty spike-train is recursively defined as: $\Delta(a \odot s) = (a + \Delta) \odot (\Delta(S))$.*

EXAMPLE *A the Δ-operator applied to a TST S with $\Delta = 1$, $S_1 =< 1, 2.1, 3 >$ and $S_2 =< 1.5, 2.1 >$, yields $1(S) = (< 2, 3.1, 4 >, < 2.5, 3.1 >)$.*

As an example we detect conjunctions BA and CD of features A, B, C, D on the input grid(fig 1B). The corresponding TST's are shown in table 1. We see that there is only output from the *cd*-detectors for the existing conjunctions. For simplicity, we took TST's with very few elements, and we left out the delay.

The proposed architecture can be implemented in a spiking neural network using large TST's. We have run experiments for complex cases, and found that the architecture can detect about 4 or 5 simultaneous *similar* conjunctions [6].

4 Conclusions

In this paper, we have defined operators that act on tuples of spike-trains. These operators can be used in an architecture that efficiently detects feature-conjunctions globally, i.e. irrespective of the number of locations on the input-grid, also in the presence of other conjunctions. Simplified feed-forward versions of the operators are implemented in spiking neural networks [6].

So far, the solutions to the perceptual binding-problem were to either create specific local feature-conjunction detectors (e.g. [7]), or to synchronize the spike-timing of neurons coding for parts of a whole (the "synchrony-hypothesis" [1]).

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 263-268

$A = (< 2.1 >, < 3.4 >)$
$B = (< 4.2 >, < 1.1 >)$
$C = (< 1.0 >, < 4.1 >)$
$D = (< 3.0 >, < 1.2 >)$

$U^L = \Sigma_i glb(U^i, U^{i+1}) = (B \lhd A) \parallel (C \lhd D) = (< 1.0 >, < 1.1 >)$
$U^R = \Sigma_i glb(U^{i+1}, U^i) = (A \lhd B) \parallel (D \lhd C) = (< 2.1 >, < 1.2 >)$

$\Sigma_i lfd(A, U^i) = A = (< 2.1 >, < 3.4 >)$
$\Sigma_i lfd(B, U^i) = B = (< 4.2 >, < 1.1 >)$
$\Sigma_i lfd(C, U^i) = C = (< 1.0 >, < 4.1 >)$
$\Sigma_i lfd(D, U^i) = D = (< 3.0 >, < 1.2 >)$

$cd(\Sigma_i lfd(B, U^i), \Sigma_i lfd(A, U^i), U^L, U^R) = (< 2.1 >, < 1.1 >)$
$cd(\Sigma_i lfd(C, U^i), \Sigma_i lfd(D, U^i), U^L, U^R) = (< 1.0 >, < 1.2 >)$
$cd(\Sigma_i lfd(C, U^i), \Sigma_i lfd(A, U^i), U^L, U^R) = (< \varepsilon >, < \varepsilon >)$
$cd(\Sigma_i lfd(B, U^i), \Sigma_i lfd(D, U^i), U^L, U^R) = (< \varepsilon >, < \varepsilon >)$

Table 1: Output of operators when only the conjunctions BA and CD are present. Detectors for BA and CD output a TST, detectors for "ghosts" CA and BD do not.

It is hard to argue that the former approach solves the problem *as posed*, and for the latter, the general feasibility is increasingly being questioned [8], also because typically neural synchronization is too slow to account for a feed-forward integration of parts into wholes. The lack of "conventional" solutions has been driving explorations into vector-based networks [4] and spiking neural networks [9, 10]: ideas that are combined in the presented framework.

References

[1] Ch. von der Malsburg. The what and why of binding: The modeler's perspective. *Neuron*, 24:95–104, 1999.
[2] J.A. Fodor and Z.W. Pylyshyn. Connectionism and cognitive architecture: a critical analysis. *Cognition*, 28:3–71, 1988.
[3] M.C. Mozer. *The Perception of Multiple Objects*. MIT Press, MA, USA, 1991.
[4] D.A. Rachkovskij and E.M. Kussul. Binding and normalization of sparse distributed representations by context-dependent thinning. *Neural Comp.*, 13:411–452, 2001.
[5] S.J. Thorpe, F. Fize, and C. Marlot. Speed of processing in the human visual system. *Nature*, 381:520–522, 1996.
[6] S.M. Bohte, J.N. Kok, and H. La Poutré. Dynamic binding in sparse spike-time vectors. in preparation, `www.cwi.nl/~sbohte`, 2001.
[7] B.W. Mel and J. Fiser. Minimizing binding errors using learned conjunctive features. *Neural Comp.*, 12:247–278, 2000.
[8] M.N. Shadlen and J.A. Movshon. Synchrony unbound: A critical evaluation of the temporal binding hypothesis. *Neuron*, 24:67–77, 1999.
[9] T. Natschläger and B. Ruf. Spatial and temporal pattern analysis via spiking neurons. *Network: Comp. Neural Syst.*, 9(3):319–338, 1998.
[10] S.M. Bohte, J.N. Kok, and H. La Poutré. Spike-prop: backpropagation for networks of spiking neurons. In M. Verleysen, editor, *Proc. ESANN'2000*, pages 419–425, 2000.