A supervised learning approach based on STDP and polychronization in spiking neuron networks

Hélène Paugam-Moisy[1], Régis Martinez[1] and Samy Bengio[2]

[1] LIRIS-CNRS - Université Lyon 2 - F-69676 Bron - France
[2] IDIAP Research Institute - CP 592, CH-1920 Martigny - Switzerland

Abstract. We propose a network model of spiking neurons, without preimposed topology and driven by STDP (Spike-Time-Dependent Plasticity), a temporal Hebbian unsupervised learning mode, biologically observed. The model is further driven by a supervised learning algorithm, based on a margin criterion, that has effect on the synaptic delays linking the network to the output neurons, with classification as a goal task. The network processing and the resulting performance are completely explainable by the concept of polychronization, proposed by Izhikevich [1]. The model emphasizes the computational capabilities of this concept.

1 Introduction

Spiking Neuron Networks (SNNs) derive their strength and interest from an accurate modeling of synaptic interactions between neurons, taking into account the time of spike emission. Although they are potentially more powerful than traditional artificial neural networks, discovering efficient learning rules adapted to SNNs is still a burning topic. A current trend is to propose computational justifications for unsupervised learning rules based on synaptic plasticity [2], especially STDP. New concepts and network architectures have been proposed, with the Echo State Network (ESN) [3] and the Liquid State Machine (LSM) [4] but learning in such networks is still hard to control.

We define a SNN model built on biological bases, with synaptic plasticity, but conceived for supervised classification. The network architecture is a sparsely connected set of neurons, without preimposed topology, as in ESN and LSM. The learning rule we propose is based on the conjunction of two main ideas: The interest of programmable delays for computational power and learnability (proved by complexity analyses, as in [5]); The notion of polychronization, as defined by Izhikevich [1] who proposes that the emergence of polychronous groups (see Section 5 for details), with persistent activation, could represent a stimulation pattern or a prototype.

The principle of our model is to adapt the delays of the network output in order to enhance the influence of the polychronous groups activated by a given pattern towards an output neuron corresponding to the pattern class.

Section 2 describes the model of SNN and Section 3 defines the two-scale learning mechanism. The performance of the model for a classification task is studied through experiments related in Section 4. Section 5 presents the notion of polychronization and explains the internal behavior of the model.

2 Spiking Neuron Network model

Network architecture The classifier is a set of M cells (internal network), interfaced with a layer of K input cells and 2 output cells, one for each class (Figure 1). The network is fed by input vectors of real numbers, represented by spikes in temporal coding: the higher the value, the earlier the spike emission towards the network. The index of the first firing output cell provides the class number, as an answer of the network to the input pattern.

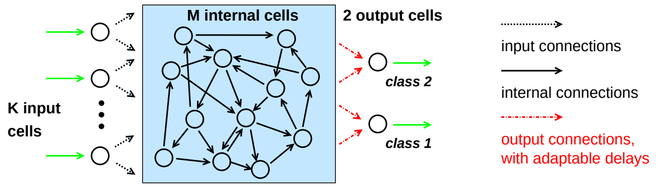


Fig. 1: Architecture of the Spiking Neuron Network as a classifier.

Each cell is a spiking neuron. Each synaptic connection, from neuron N_i to neuron N_j, is defined by a weight w_{ij} and a transmission delay d_{ij}. The internal network is composed of 80% excitatory neurons and 20% inhibitory neurons. The connectivity is random and sparse with probability 0.3 inside the internal network. For pattern stimulation, the input cells are connected to the internal cells with probability 0.1. For class detection, the internal cells are fully connected to each output neuron.

Model of Neuron The neuron model is an SRM_0 ("Spike Response Model"), as defined in [6], where the state of a neuron N_j is dependent on its last spike time $t_j^{(f)}$ only. The next firing time of N_j is governed by its membrane potential $u_j(t)$ and its threshold $\theta_j(t)$. Both variables are functions of the last firing times of the neurons N_i belonging to the set Γ_j of neurons presynaptic to N_j:

$$u_j(t) = \underbrace{\eta(t - t_j^{(f)})}_{threshold\ kernel} + \sum_{i \in \Gamma_j} w_{ij} \underbrace{\epsilon(t - t_i^{(f)} - d_{ij})}_{potential\ kernel} \quad \text{and} \quad u_j(t) \geq \vartheta \Longrightarrow t_j^{(f+1)} = t \tag{1}$$

where the potential kernel is modelled by a Dirac increase in 0, followed by an exponential decrease, from value $u_{max} = 8mV$ in 0^+ towards 0, with a time constant $\tau_m = 2ms$. The firing threshold ϑ is set to $-50mV$ and the threshold kernel simulates an absolute refractory period $\tau_{abs} = 7ms$, when the neuron cannot fire again, followed by a reset to the resting potential $u_{rest} = -65mV$. Simulations are computed in discrete time, with $1ms$ steps.

Synaptic Plasticity Each synaptic weight w_{ij} can be adapted by STDP (Spike-Time-Dependent Plasticity), a form of synaptic plasticity based on the respective order of pre- and postsynaptic firing times. Basically, a causal order (pre- just before post-) strengthens the connection whereas a non-causal order decreases its weight. A STDP temporal window is a function used to calculate the weight modification ΔW as a function of the time difference $\Delta t = t_{post} - t_{pre} = t_j^{(f)} - (t_i^{(f)} + d_{ij})$ and can be computed at the level of neuron N_j. We use excitatory and inhibitory temporal windows as proposed in [7] and apply a multiplicative weight update.

3 Learning Mechanism

There are two concurrent learning mechanisms in the model: An unsupervised learning of weights by STDP, operating in the millisecond range, at each new impact t_{pre} or emission t_{post} of a spike, and a supervised learning of output delays, operating in the range of $100ms$, at each pattern presentation.

STDP Implementation STDP is applied to the weights of internal cells only. The other weights of connections are kept fixed, with value $w_{IN} = 3$ from input layer to internal network and value $w_{OUT} = 0.5$ from internal network to output neurons. The delays d_{ij} take integer values, randomly chosen in $\{1, \ldots, 20\}$, both in the internal network and towards output neurons. Delays from input cells are set to zero, for an immediate transmission of input information. We switch to machine learning in designing a supervised mechanism, based on a margin criterion, for adapting the output delays of the model.

Delay Adaptation Algorithm The goal of the supervised learning mechanism we propose is to modify the delays from active internal neurons to output neurons in such a way that the output neuron corresponding to the target class fires before the one corresponding to the non-target class. Moreover, as it has been shown in the machine learning literature, maximizing a margin between the positive and the negative class yields better expected generalization performance [8]. More formally, we thus try to minimize the following criterion:

$$C = \sum_{p \in \text{class1}} |t_1(p) - t_2(p) + \epsilon|_+ + \sum_{p \in \text{class2}} |t_2(p) - t_1(p) + \epsilon|_+ \tag{2}$$

where $t_i(p)$ represents the firing time of output neuron i answering to input pattern p, ϵ represents the minimum delay margin we want to enforce between the two firing times, and $|z|_+ = \max(0, z)$. In order to minimize this criterion, we adopt a stochastic training approach, iterating the loop:

```
After the presentation of a given input pattern p,
if difference of firing times between target and non-target output neuron < ε,
then for each output neuron, select the connection that received the decisive
impact, among presynaptic neurons responsible for the output spike,
• for target neuron :  decrement the delay (−1ms)
• for non-target neuron :  increment the delay (+1ms)
```

Hence, at each step, we decrease the probability of an error in the next answer to a similar input and we help for a larger time range between the 2 output firing.

4 Classifier Performance

Experiments After an initialization phase generating a high disordered activity, a learning phase is run, from 2000 to $11000ms$ (in simulated biological time), with successive alternated presentations of two input patterns, similar to Izhikevich stimulation patterns (Fig. 12 in [1]). Two oblique bars represent examples for class 1 and class 2 respectively (bottom of Figure 2, left).

A spike raster plot presents all the firing times of all the neurons: neuron index with respect to time (in ms), for $K = 10$ input neurons (bottom), followed by $M = 100$ internal neurons (including 20 inhibitory neurons). Firing times of the two output neurons are isolated at the top.

Afterwards, a generalization phase is run, with noisy patterns: Each spike time occurs at $t \pm \eta$ where t is the firing time of the corresponding input neuron for the example pattern of the same class and η is some uniform noise.

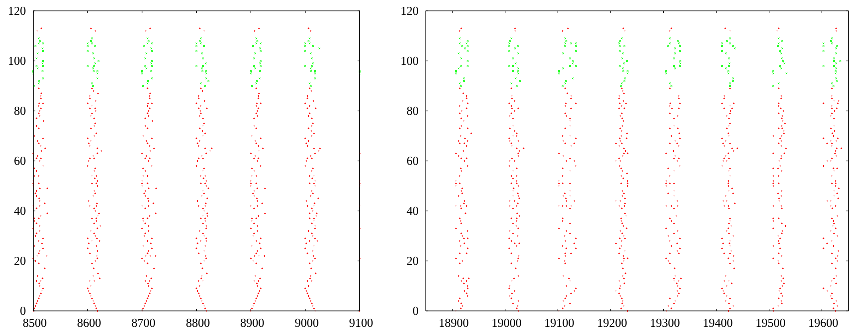


Fig. 2: Spike raster plots, near the end of the learning phase (left) and during generalization phase, with noisy inputs, $\eta = 8$ (right).

Results As can be observed on Figure 2 (left), the internal network activity stabilizes on a persistent alternative between two different spike-timing patterns, one for each class. The firing times of the two output neurons reflect the application of the delay adaptation algorithm: Starting from simultaneous firing (not shown), step by step, they slowly dissociate their responses, becoming selective to the class the input belongs to. In the left frame of Figure 2, the time interval separating the two output spikes has become stable, since the margin ϵ has been reached.

Slight variations in network activity are due to the still running STDP adaptation of weights. At the end of the learning phase, half the excitatory weights are distributed from 0.1 to 0.7, the other being close to 0. Inhibitory weights are widely distributed, mainly between 0.5 and 0.9, and around 0.

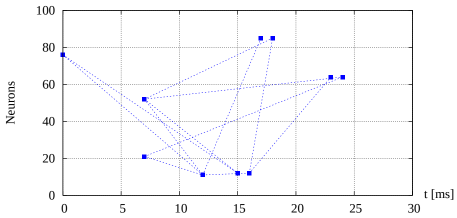
In generalization phase, although the internal network activity is clearly disrupted, the classification performance remains good: average success rate, on 100 noisy patterns of each class, is 96% for $\eta = 4$, and still 81% for $\eta = 8$ (Fig. 2, right), where the input patterns are hard to discriminate by a human observer.

5 Polychronization

The links between high-level cognitive processes (active perception, memorization via neuron assemblies) and the dynamics of spike timing inside natural or artificial neural networks is presently an active research area. Several hypotheses have been proposed, mainly based on synchronization (e.g. synfire chain [9]).

Polychronous Groups The hypothesis of synchronization is too much restrictive when it comes to grasp the full power of neuron assemblies processing. Instead, Izhikevich [1] proposes the notion of polychronization. Based on the connectivity between neurons, a polychronous group is a possible stereotypical time-locked firing pattern. Since neurons of a polychronous group have matching axonal conduction delays, the group can be the basis of a reproducible spike-timing pattern: Firing of the first few neurons with the right timing is enough to activate most of the group. As any neuron can be activated within several polychronous groups, at different times, the number of coexisting polychronous groups in a network can be very huge, thus opening possibility of high memory capacity.

In our model, all the potential polychronous groups in the internal network, depending on its topology and the values of the delays, can be inventoried. We have detected 104 potentially activable polychronous groups in a network of $M = 100$ neurons, and more than 3000 in a network of $M = 200$ neurons.



Triggering neurons: 21,52,76, with timing {7,7,0}

Fig. 3: An example of polychronous group: Starting from the three initial triggering neurons, further neurons of the group can be activated, in chain, with respect to the spike-timing pattern represented on the diagram.

Our model proposes a way to confirm the link between a correct identification of an input presentation and persistent spike-timing patterns inside the network.

Learning Process Justification First, we observed (data not shown) that most of the neurons responsible for delay adaptation belong to active polychronous groups. Second, we analysed the evolution of the polychronous groups activation at two stages of the learning phase (Figure 4). The two frames show the ratio of activation (in percentage) of a representative sample of polychronous groups (# 60 to 100) in response to each input pattern : black for class 1 and grey for class 2. A large amount of polychronous groups are activated at the beginning of learning (left), while less groups are activated at the end of the simulation (right). This can be considered as a consequence of regulation generated by STDP: Only a few polychronous groups are selected to represent the input patterns. We also observe that polychronous groups specialize for one particular class (right) instead of answering for both of them, as they did first (left).

Groups 67 and 95 perfectly illustrate the phenomenon. This observation validates that synaptic plasticity provides the network with valuable adaptability and highlights the importance of combining STDP with delay learning.

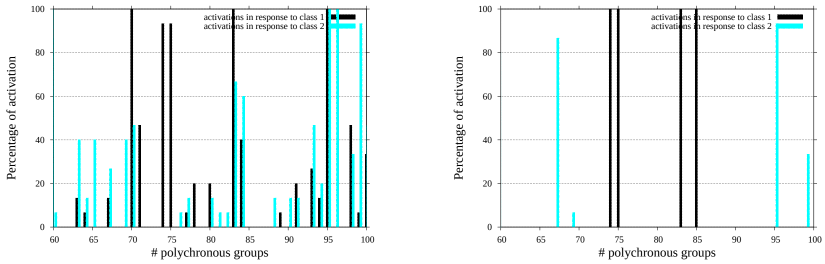


Fig. 4: Activation ratio from 2000 to 5000ms, and then from 8000 to 11000ms.

6 Conclusion

With supervised classification as a goal task, we have proposed a two-scale learning mechanism for SNNs, with unconstraint topology. The algorithm for delay adaptation is computationally easy to implement. Moreover, the way the learning algorithm operates can be well explained by the concept of polychronization and the internal network is no longer a black-box, contrary to the ESN or LSM models. The model has shown promising performance for learning and generalization on a classification task. This latter point is currently under investigation in larger scale experiments, from OCR benchmark: first trials on a two-class version of the USPS digit database yielded encouraging results.

References

[1] E.M. Izhikevich. Polychronization: Computation with spikes. *Neural Computation*, 18(1):245–282, 2006.

[2] L.F. Abbott and S.B. Nelson. Synaptic plasticity: taming the beast. *Nature Neuroscience*, 3:1178–1183, 2000.

[3] H. Jaeger. Adaptive nonlinear system identification with Echo State Networks. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*2002, Advances in Neural Information Processing Systems*, volume 15, pages 593–600. MIT Press, 2003.

[4] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[5] W. Maass and M. Schmitt. On the complexity of learning for a spiking neuron. In *COLT'97, Conf. on Computational Learning Theory*, pages 54–61. ACM Press, 1997.

[6] W. Gerstner and W. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity.* Cambridge University Press, 2002.

[7] D. Meunier and H. Paugam-Moisy. Evolutionary supervision of a dynamical neural network allows learning with on-going weights. In *IJCNN'2005, Int. Joint Conf. on Neural Networks*, pages 1493–1498. IEEE–INNS, 2005.

[8] V.N. Vapnik. *Statistical learning theory.* Wiley, 1998.

[9] M. Abeles. *Corticonics: Neural Circuits of the Cerebral Cortex.* Cambridge Press, 1991.