Tempotron-like learning with ReSuMe

Răzvan V. Florian

[1] Center for Cognitive and Neural Studies (Coneural),
Str. Cireşilor nr. 29, 400487 Cluj-Napoca, Romania
`florian@coneural.org`
[2] Babeş-Bolyai University, Department of Computer Science,
Str. Al. Kogălniceanu nr. 1, 400084 Cluj-Napoca, Romania

Abstract. The tempotron is a model of supervised learning that allows a spiking neuron to discriminate between different categories of spike trains, by firing or not as function of the category. We show that tempotron learning is quasi-equivalent to an application for a specific problem of a previously proposed, more general and biologically plausible, supervised learning rule (ReSuMe). Moreover, we show through simulations that by using ReSuMe one can train neurons to categorize spike trains not only by firing or not, but also by firing given spike trains, in contrast to the original tempotron proposal.

Published in V. Kurkova, R. Neruda, J. Koutnik (eds.), *Artificial Neural Networks - Proceedings of the 18th International Conference (ICANN 2008), Prague, Czech Republic*, Lecture Notes in Computer Science 5164, pp. 368–375, Springer Verlag, 2008. DOI: 10.1007/978-3-540-87559-8_38

1 Introduction

The tempotron has been recently proposed as a "new, biologically plausible supervised synaptic learning rule that enables neurons to efficiently learn a broad range of decision rules, even when information is embedded in the spatiotemporal structure of spike patterns rather than in mean firing rates" [1]. A few other supervised rules for spiking neurons have been previously proposed (for a review, see [2]). Here we show that a particularization of ReSuMe, one of those rules [3,4,5], is quasi-equivalent to the tempotron. Moreover, ReSuMe allows the training of tempotrons that are able to fire specific spike patterns in response to each input category.

2 The Tempotron

The tempotron learning rule [1] can be applied to a spiking neuron driven by synaptic afferents. The learning rule modifies the efficacies of the afferent synapses such that the trained neuron emits one spike when presented with inputs corresponding to one category and no spike when the inputs correspond to another category. The tempotron setup assumes that, before being presented with an input spike train, the neuron's potential is at rest, and that after the neuron emits a spike in response to an input pattern all other incoming spikes are shunted and have no effect on the neuron. Thus, even if the neuron would fire more than one spike, the spikes following the first one are artificially eliminated.

The subthreshold membrane voltage u of the trained neuron is modeled as a sum of postsynaptic potentials:

$$u(t) = u_0 + \sum_i w_i \sum_{t_i^f < t} \epsilon(t - t_i^f), \tag{1}$$

where u_0 is the resting potential, w_i is the synaptic efficacy of synapse i, and $\epsilon(t - t_i^f)$ describes the form of the postsynaptic potential induced in the neuron by a spike at t_i^f received from neuron i. The first sum runs over all presynaptic neurons, and the second one runs over all spikes of neuron i prior to t. When u overcomes the firing threshold θ, the neuron emits a spike.

Tempotron learning minimizes the following cost function, for each input pattern [1]:

$$C = \begin{cases} \theta - u_{\max}, & \text{if } u_{\max} < \theta \text{ and the neuron should fire for this pattern,} \\ u_{\max} - \theta, & \text{if the neuron fired } (u_{\max} \geq \theta) \text{ and it should have been silent,} \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $u_{\max} = u(t_{\max})$ is the maximal value of the postsynaptic potential u, in the case that the neuron did not fire. In the case that the neuron fired, $u_{\max}$ is the maximal value that u would have been reached if the neuron would have not fired.

Applying the gradient descent method in the space of synaptic efficacies for minimizing the above cost function leads to the tempotron learning rule [1]:

$$\Delta w_i = \begin{cases} \lambda \sum_{t_i^f < t_{\max}} \epsilon(t_{\max} - t_i^f), & \text{if the neuron should fire but it did not,} \\ -\lambda \sum_{t_i^f < t_{\max}} \epsilon(t_{\max} - t_i^f), & \text{if the neuron should not fire but it did,} \\ 0, & \text{otherwise,} \end{cases} \tag{3}$$

where $\lambda > 0$ is the learning rate. During learning, synaptic changes Δw_i are applied after each presentation of an input pattern.

It can be seen that the dynamics of this learning rule has little biological plausibility, since: a) It requires the monitoring of the maximum of u; b) For trials when the neuron fires while it should not, it requires, for computing $t_{\max}$, simulating a dynamics of the neuron that is different from the real one (since it ignores that the neuron fired and the membrane potential was reset). The setup also has little biological plausibility, since: a) While it assumes that the precision of spike times in input patterns is important, it ignores the time when the trained neuron fires. If the coding of information in the brain depends on the precision of spike times, as experimental studies have suggested [6], then the timing of the firing of both the afferents and the trained neurons should matter;

b) It is assumed that the trained neuron can fire either no spike or just one spike per input pattern; c) It is assumed that learned input patterns are isolated from other inputs and thus that the trained neuron is initially at rest, which is not the case in the brain.

3 ReSuMe

The ReSuMe learning rule [3,4,5,7,8] also allows the supervised training of a neuron and is defined by the following equation:

$$\frac{\mathrm{d}w_i(t)}{\mathrm{d}t} = \lambda \, [\tilde{\Phi}(t) - \Phi(t)] \left[a + \int_0^\infty W(s)\, \Phi_i(t-s)\, \mathrm{d}s \right], \tag{4}$$

where $\tilde{\Phi}(t) = \sum_{f=1}^{\tilde{n}} \delta(t - \tilde{t}^f)$ is the target spike train to be learned by the neuron, represented by a sum of Dirac pulses; $\Phi(t) = \sum_{f=1}^{n} \delta(t - t^f)$ is the actual output of the neuron; $\Phi_i(t)$ is the input spike train coming from synapse i, also a sum of Dirac pulses; a is a constant; $\tilde{t}^f$ are the moments of spikes in the target spike train and $\tilde{n}$ their number; t^f are the moments of spikes in the actual output spike train and n their number; and W is a learning window that was originally proposed to be $W(s) = E(s)$ with

$$E(s) = A \, \exp(-s/\tau_E) \tag{5}$$

where A and τ_E are positive constants. It can be seen that, after a learning trial, the synaptic change is

$$\Delta w_i = \lambda \, a \, (\tilde{n} - n) + \lambda \sum_{\tilde{t}^g} \sum_{t_i^f \leq \tilde{t}^g} W(\tilde{t}^g - t_i^f) - \lambda \sum_{t^g} \sum_{t_i^f \leq t^g} W(t^g - t_i^f). \tag{6}$$

4 Applying ReSuMe to the Tempotron Problem

By applying the ReSuMe rule to the tempotron setup (one target output spike or none, one output spike at t^1 or none), and if we note that for having one output spike, regardless of its timing, it is the easiest to have it at $\tilde{t}^1 = t_{\max}$, if the neuron did not fire, or at the actual time of firing $\tilde{t}^1 = t^1$, if it fires, we get:

$$\Delta w_i = \begin{cases} \lambda \, a + \lambda \sum_{t_i^f \leq t_{\max}} W(t_{\max} - t_i^f), & \text{if } \tilde{n} = 1, n = 0, \\ -\lambda \, a - \lambda \sum_{t_i^f \leq t^1} W(t^1 - t_i^f), & \text{if } \tilde{n} = 0, n = 1, \\ 0, & \text{if } \tilde{n} = n. \end{cases} \tag{7}$$

In the first and the lase case in the last equation, the ReSuMe learning rule is equivalent to the tempotron learning rule for $a = 0$ and $W(s) = \epsilon(s)$. In the second case, the learning rules can be considered equivalent if we note that, if the trained neuron fires, the maximum of u, θ, is actually reached at the firing time t^1, and this is the closest approximation to $t_{\max}$ that can be made with quantities locally available to the neuron. The postsynaptic potential ϵ can be well approximated by the exponential E that was originally used for ReSuMe; for example, for integrate-and-fire neurons with postsynaptic currents that are Dirac pulses, $\epsilon(s)$ is exactly an exponential. Thus, the tempotron learning rule can be considered a particular application of ReSuMe for a particular problem.

But ReSuMe is a more general and more biologically plausible learning rule, since, in contrast to the tempotron: a) If the trained neuron is assumed to fire in response to a pattern, one can not only teach it to fire, but also to fire at particular moments, which is more biologically plausible and also permits a finer control of the neuron's behavior; b) It allows discriminating between more than two input categories; c) It does not require monitoring the maximum of u; d) It allows not only episodic learning but also online learning. Of course, the biological plausibility of supervised learning rules for spiking neural networks is limited by the constraint of providing a teaching signal for each considered neuron.

Because the tempotron learning rule minimizes the cost function defined by Eq. 2 and of the quasi-equivalence of ReSuMe with the tempotron learning rule for the tempotron problem, ReSuMe, with $W(s) = \epsilon(s)$, is an optimal learning rule for the tempotron problem. In the case that, additionally to the tempotron setup, we would like to teach the neuron to fire at a particular moment in time, it is currently known that ReSuMe, with $W(s) = E(s)$ will converge to an optimal solution at least for the case of one input with one spike and one target output spike [8].

5 Simulations

In order to verify the assertions presented above, we performed several simulations. We implemented the setup for learning to classify latency patterns, which was originally used for demonstrating the efficacy of the tempotron learning rule [1]. The trained neuron must learn to separate p input patterns into two categories. For each category, the neuron has to have a distinct, characteristic output. The input patterns are generated randomly and are assigned randomly to one of the two categories. Each input pattern has a duration $T = 500$ ms and consists of one spike for each of the $N = 500$ afferent synapses of the trained neuron. The timing of each of these spikes is generated randomly with uniform distribution between 0 and T. Except where specified, the parameters of the simulation are as in [1]. The trained neuron is an integrate-and-fire neuron with the time constant of the membrane decay τ. Each input spike generates an exponentially decaying current with time constant τ_s. Thus, for the case that the neuron has not yet emitted a spike,

$$\epsilon(t - t_i^f) = \epsilon_0 \left[\exp\left(-\frac{t - t_i^f}{\tau}\right) - \exp\left(-\frac{t - t_i^f}{\tau_s}\right) \right], \tag{8}$$

as in [1]. For the results presented here, we use $p = 50$ input patterns.

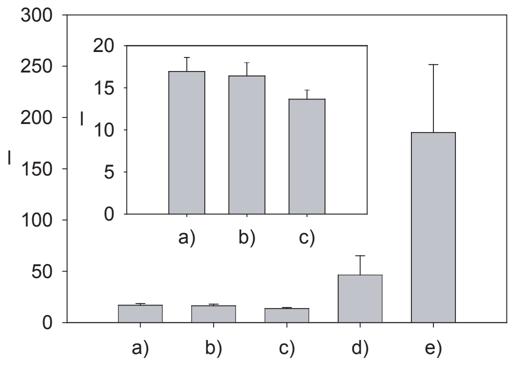

Fig. 1. The number l of learning trials needed for perfect learning of classifying latency patterns, for various implementation of the learning rule and various problems. For each rule / setup, averages and standard deviations are computed over 100 experiments with different, random initial conditions. a) Original tempotron learning rule. b) Modified tempotron learning rule ($t_{\max}$ replaced by t^1 when the output neuron fires), equivalent to the ReSuMe learning rule with $a = 0$ and $W(s) = \epsilon(s)$. c) ReSuMe learning rule for the tempotron setup (with $W(s) = E(s)$). d) Learning with ReSuMe of a classification task where, when the neuron fires, it has to fire at a particular moment in time. e) Learning with ReSuMe of a classification task where, for both categories, the neuron has to fire one spike at particular moments in time. See text for details. Inset: zoom over the results of a), b), c).

In the original tempotron setup, the two categories are named $\ominus$ and $\oplus$ and the trained neuron has to fire no spike for the $\ominus$ patterns and one spike, regardless of its timing, for the $\oplus$ patterns. Our first simulations implemented the original tempotron learning rule for this setup and checked whether the changes suggested by the application of ReSuMe to the tempotron setup affect the efficacy of learning. In the simulations, we trained the neuron until there was no error (all patterns were classified correctly) and recorded the number l of learning trials needed for learning. A trial consists of presentations of each of the p patterns, followed by applying the changes of w_i given by the learning rule.

We first reproduced learning with the original tempotron rule (Eq. 3; Fig. 1a). We then performed the simulation by replacing $t_{\max}$ in Eq. 3, for the cases where the output neuron spiked, with the actual timing of the output spike t^1. Thus, we effectively implemented the ReSuMe learning rule with $a = 0$ and $W(s) = \epsilon(s)$. The results are presented in Fig. 1b and show that there is no increase in learning time, for equal performance.

We then ran the simulation by using an exponential learning window, $W(s) = E(s)$ (Eq. 5), as originally proposed for the ReSuMe rule, with $A = 1$ and $\tau_E = \tau$. Again, we can still achieve learning with no errors, and learning time for this method is even better than for the tempotron (Fig. 1c). This is consistent with other results obtained in simulations, that have shown that ReSuMe has better performance with an exponential learning window (E, Eq. 5) than with a double-exponential learning window (like ϵ, Eq. 8) [9]. This is, however, somehow surprising, given current theoretical understanding of ReSuMe: it has been shown analytically that ReSuMe with $W(s) = E(s)$ converges to the solution only for one input [8] (but we use here multiple inputs); while the tempotron learning rule performs gradient descent towards the solution [1].

We then explored the more difficult, but more general and relevant problem where we remove the artificial shunting of the inputs after the output neuron fires (thus allowing it to fire more than one spike) but still require the neuron to learn to fire only one spike for the $\oplus$ patterns. Thus, there will be one output spike for the $\oplus$ patterns not because the number of spikes is restricted artificially but because the neuron has to adapt its synapses in order to do so. Moreover, we require the neuron to fire this spike at precisely $\tilde{t}^1 = 400$ ms after the beginning of an input pattern. As previously, the neuron has to fire no spike for $\ominus$ patterns. For this, we use the general ReSuMe learning rule, Eq. 6, with $a = 0$, $W(s) = E(s)$, $A = 1$, and $\tau_E = \tau$, and we keep training the neuron until the number of output spikes is the desired one for each input pattern. We now also have to consider the dynamics of the trained neuron after it emits a spike. We do this according to the standard integrate-and-fire model [10], and in this case we have, when the timing of a presynaptic spike t_i^f precedes the timing of the latest postsynaptic spike $\hat{t}$,

$$\epsilon(t - t_i^f) = \epsilon_0 \exp\left(-\frac{\hat{t} - t_i^f}{\tau_s}\right)\left[\exp\left(-\frac{t - \hat{t}}{\tau_m}\right) - \exp\left(-\frac{t - \hat{t}}{\tau_s}\right)\right], \qquad (9)$$

while when $t_i^f > \hat{t}$ the form of ϵ from Eq. 8 is still valid. We used a reset potential equal to the rest potential, so there was no refractory kernel. Again, we achieve learning, and even if the problem is significantly more difficult the learning time is only about 3 times higher than for the simpler tempotron problem (Fig. 1d). As it can be seen in Fig. 2a, the time at which the neuron fires after learning in response to $\oplus$ patterns is very narrowly distributed around $\tilde{t}^1$.

Finally, we tackled the still more difficult problem of having the neuron fire one spike at $\tilde{t}^1_\oplus = 350$ ms in response to a $\oplus$ pattern and one spike at $\tilde{t}^1_\ominus = 450$ ms in response to a $\ominus$ pattern, in the same conditions as for the previous simulation. We still achieve learning, after a larger number of training epochs (Fig. 1e), and the distribution of the moments at which the neuron fires after learning is again narrowly distributed around the desired times (Fig. 2b,c).

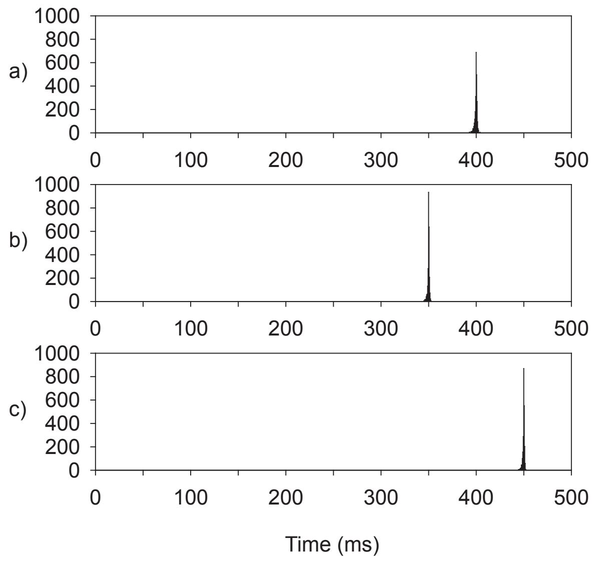

Fig. 2. Histograms of the distribution of spike timings of the output neuron, after learning, when responding to input spike pattern categories. The distribution contains the responses of the neuron to each pattern within a category (about $p/2 = 250$ patterns per category), for each trial, and for 100 trials with random initial conditions. Bin width is 0.5 ms. a) The neuron has to either fire a spike at $\tilde{t}^1 = 400$ ms in response to one category or not to fire at all. b), c) The neuron has to fire one spike at $\tilde{t}^1_\oplus = 350$ ms in response to a $\oplus$ pattern (b) and one spike at $\tilde{t}^1_\ominus = 450$ ms in response to a $\ominus$ pattern (c).

6 Conclusions

We have demonstrated the equivalence between tempotron and ReSuMe, under certain conditions, and we have shown that ReSuMe is a more general and more biologically-plausible approach to supervised learning for spiking neurons than the tempotron. The tempotron learning rule is, in fact, a particular case of the ReSuMe learning rule for a specific, quite artificial problem. Moreover, we have shown in simulations that by using the ReSuMe learning rule one can train neurons to classify input patterns not only by indicating the class by firing or not firing in a given time interval, but also by firing spikes with precise timings.

If one considers that representing information in the precise spike timings is relevant, as it was considered for the input spike trains in the tempotron setup, then the output of spike train classifiers should also be capable of representing information temporally. Hence general learning rules that are capable of learning spike times, such as ReSuMe, should be used instead of the tempotron learning rule for such problems. Having the same type of coding for both input and output permits using the output of a classifier as the input of another similar classifier, thus forming networks with higher information processing capabilities. A spiking classifier with temporally coded output is also important for reservoir computing [11] when using a spiking reservoir. In this case, one may train a spiking readout and feed its output back into the reservoir (since information is coded as in the reservoir), thus improving the computational power of the network [12].

Despite its limitations, the tempotron has been proved quite efficient for spoken digit recognition, outperforming with only 15 spiking neurons complex state-of-the-art Hidden Markov Model word recognition systems [13]. This shows that spiking neural networks are quite powerful, and using appropriate learning methods for training them might reveal even more of their potential.

Acknowledgements. This work has been sponsored by grants of the Romanian National Authority for Scientific Research (PNCDI II, Parteneriate, contract no. 11-039/2007; CEEX, contract no. 1474/2006). I thank to Filip Ponulak and Raul C. Mureşan for useful suggestions.

References

1. Gütig, R., Sompolinsky, H.: The tempotron: a neuron that learns spike timing-based decisions. Nature Neuroscience **9**(3) (2006) 420–428
2. Kasiński, A., Ponulak, F.: Comparison of supervised learning methods for spike time coding in spiking neural networks. International Journal of Applied Mathematics and Computer Science **16**(1) (2006) 101–113
3. Ponulak, F.: ReSuMe — new supervised learning method for the spiking neural networks. Technical report, Institute of Control and Information Engineering, Poznań University of Technology, Poland (2005)
4. Kasiński, A., Ponulak, F.: Experimental demonstration of learning properties of a new supervised learning method for the spiking neural networks. In Duch, W., Kacprzyk, J., Oja, E., Zadrozny, S., eds.: Proceedings of the 15th International Conference on Artificial Neural Networks: Biological Inspirations. Volume 3696 of Lecture Notes in Computer Science. Springer (2005) 145–152
5. Ponulak, F.: Supervised learning in spiking neural networks with ReSuMe method. PhD thesis, Poznań University of Technology, Poland (2006)
6. Bohte, S.M.: The evidence for neural information processing with precise spike-times: A survey. Natural Computing **3**(2) (2004) 195–206
7. Ponulak, F., Kasiński, A.: Generalization properties of SNN trained with ReSuMe. In: Proceedings of the European Symposium on Artificial Neural Networks, ESANN'2006, Bruges, Belgium. (2006)
8. Ponulak, F.: ReSuMe — proof of convergence. Technical report, Institute of Control and Information Engineering, Poznań University of Technology, Poland (2006)
9. Ponulak, F.: Analysis of ReSuMe learning process for spiking neural networks. International Journal of Applied Mathematics and Computer Science **18**(2) (2008) 117–127
10. Gerstner, W., Kistler, W.M.: Spiking neuron models. Cambridge University Press, Cambridge, UK (2002)

11. Schrauwen, B., Verstraeten, D., Van Campenhout, J.: An overview of reservoir computing: theory, applications and implementations. In: Proceedings of the 15th European Symposium on Artificial Neural Networks. (2007) 471–482
12. Maass, W., Joshi, P., Sontag, E.D.: Computational aspects of feedback in neural circuits. PLoS Computational Biology **3**(1) (2007) e165
13. Gütig, R., Sompolinsky, H.: Neural mechanisms of speech processing: time warp invariants in adaptive integration time. In: Cosyne 2008: Computational and Systems Neuroscience. (2008) 337