Classification of Distorted Patterns by Feed-forward Spiking Neural Networks

Ioana Sporea and André Grüning

University of Surrey, Department of Computing,
Guildford, GU2 7XH, United Kingdom
{i.nica,a.gruning}@surrey.ac.uk
www.surrey.ac.uk

Abstract. In this paper, a feed forward spiking neural network is tested with spike train patterns with additional and missing spikes. The network is trained with noisy and distorted patterns with an extension of the ReSuMe learning rule to networks with hidden layers. The results show that the multilayer ReSuMe can reliably learn to discriminate highly distorted patterns spanning over 500 ms.

Keywords: error backpropagation, multilayer ReSuMe, distorted spike train patterns, spiking neurons

1 Introduction

Artificial networks of rate coded neurons have been successfully used in many classification problems, however these are mostly limited to spatial patterns. Thus, these models are ill-suited to discriminate even simple temporal patterns. Their biological plausibility is now also challenged by experimental findings regarding the importance of the relative timing of individual spikes [15, 9, 2]. These findings have led to a new way of simulating neural networks based on temporal encoding with single spikes [10].

Learning algorithms are essential in addressing any pattern classification problem. However, few supervised learning algorithms exists for networks of spiking neurons and little is known about their suitability for real-world problems where data is predominantly noisy and distorted.

In this paper, we present the capability of a new supervised learning algorithm, multilayer ReSuMe [11], to train feed-forward spiking neural networks using noisy and distorted temporal patterns and the network's ability to correctly classify highly distorted patterns after learning.

2 Multilayer ReSuMe

In this section, we restate briefly the derivation of a supervised learning algorithm, multilayer ReSuMe, from [11] that extends the ReSuMe learning rule [12] to feed-forward networks with hidden layers.

The learning algorithm is derived for a fully connected feed-forward network with hidden layers (see [11] for a detailed derivation). All neurons in one layer are connected to all neurons in the subsequent layer. The input and output signals of spiking neurons are represented by the timing of spikes. A spike train is defined as a sequence of impulses fired by a particular neuron at times t^f. Spike trains are formalised by a sum of Dirac δ functions: $S(t) = \sum_f \delta(t - t^f)$ [4].

The spiking neurons are modelled by the linear Poisson neuron model [5, 8]. Its instantaneous firing rate $R(t)$ is formally defined as the expectation of the spike train, averaged over an infinite number of trials. An estimate of the instantaneous firing rate can be obtained by averaging over a finite number of trials [7]:

$$R(t) =< S(t) >= \frac{1}{M} \sum_{j=1}^{M} S_j(t) \tag{1}$$

where M is the number of trials and $S_j(t)$ is the concrete spike train for each trial. The instantaneous firing rate $R(t)$ will be replaced by its discontinuous estimate, the spike train $S(t)$.

The instantaneous network error is formally defined in terms of the difference between the actual instantaneous firing rate $R_o^a(t)$ and the target instantaneous firing rate $R_o^d(t)$ for all output neurons:

$$E(t) = E(R_o^a(t)) = \frac{1}{2} \sum_{o \in O} \left(R_o^a(t) - R_o^d(t) \right)^2 \tag{2}$$

In order to minimise the network error, the weights are modified using a process of gradient descent:

$$\Delta w_{oh}(t) = -\frac{\partial E(R_o^a(t))}{\partial w_{oh}} \tag{3}$$

where w_{oh} represents the weight between the output neuron o and hidden neuron h. $\Delta w_{oh}(t)$ is the weight change contribution due to the error $E(t)$ at time t, and the total weight change is $\Delta w = \int \Delta w(t) dt$ over the duration of the spike train.

The weights are modified similarly to standard back-propagation for rate neurons using the back-propagation error $\delta_o(t)$:

$$\Delta w_{oh}(t) = \delta_o(t) R_h(t) \tag{4}$$

This is similar to standard discrete-time backpropagation, however now derived as a functional derivative in continuous time. The back-propagation error is defined as follows:

$$\delta_o(t) := \frac{1}{n_h} \left[R_o^d(t) - R_o^a(t) \right] \text{, for output neurons} \tag{5}$$

where n_h is the number of hidden neurons.

$$\delta_h(t) := \frac{1}{n_i} \sum_{o \in O} \delta_o(t) |w_{oh}| \text{, for hidden neurons} \tag{6}$$

where n_i is the number of input neurons and $\delta_o(t)$ is the back-propagated error of the previous layer.

In the following we will use the best estimation of the unknown instantaneous firing rate $R(t)$ when we only have a single spike train $S(t)$, which is the spike train itself for each of the neurons involved. Thus the weights will be modified according to:

$$\Delta w_{oh}(t) = \frac{1}{n_h}\left[S_o^d(t) - S_o^a(t)\right] S_h(t) \tag{7}$$

However, products of Dirac δ functions are mathematically problematic. Following [12] the non-linear product of $S_o^d(t)S_h(t)$ is substituted with a STDP process. In a similar manner, $S_o^a(t)S_h(t)$ is substituted with an anti-STDP process (for details see [12]).

$$\begin{aligned} S_o^d(t)S_h(t) \rightarrow & S_h(t)\left[a + \int_0^\infty a^{pre}(s)S_o^d(t-s)ds\right] \\ & + S_o^d(t)\left[a + \int_0^\infty a^{post}(s)S_h(t-s)ds\right] \end{aligned} \tag{8}$$

$$\begin{aligned} S_o^a(t)S_h(t) \rightarrow & - S_h(t)\left[a + \int_0^\infty a^{pre}(s)S_o^a(t-s)ds\right] \\ & - S_o^a(t)\left[a + \int_0^\infty a^{post}(s)S_h(t-s)ds\right] \end{aligned} \tag{9}$$

where $a > 0$ is a non-Hebbian term that guarantees the weight changes in the correct direction if the output spike train contains more or less spikes than the target spike train.

The integration variable s represents the time difference between the actual firing time of the output neuron and the firing time of the hidden neuron $s = (t_o^f - t_h^f)$, and the target firing time and the firing time of the hidden neuron $s = (t_d^f - t_h^f)$ respectively. The kernel $a^{pre}(s)$ gives the weight change if the presynaptic spike (the spike of the hidden neuron) comes after the postsynaptic spike (the spikes of the output and target neurons). The kernel $a^{post}(s)$ gives the weight change if the presynaptic spike before the postsynaptic spike. The kernels a^{pre} and a^{post} define the learning window $W(s)$ [4]:

$$W(s) = \begin{cases} a^{pre}(-s) = -A_- \exp(\frac{s}{\tau_-}), & \text{if } s \leq 0 \\ a^{post}(s) = +A_+ \exp(\frac{-s}{\tau_+}), & \text{if } s > 0 \end{cases} \tag{10}$$

where A_+, $A_- > 0$ are the amplitudes and τ_+, $\tau_- > 0$ are the time constants of the learning window. Thus the final learning formula for the weight modifications for output neurons becomes:

$$\begin{aligned} \Delta w_{oh}(t) = & \frac{1}{n_h} S_h(t)\left[\int_0^\infty a^{pre}(s)[S_o^d(t-s) - S_o^a(t-s)]ds\right] \\ & + \frac{1}{n_h}\left[S_o^d(t) - S_o^a(t)\right]\left[a + \int_0^\infty a^{post}(s)S_h(t-s)ds\right] \end{aligned} \tag{11}$$

For a network with a single hidden layer, the weights for the hidden neurons are updated according to:

$$\begin{aligned}\mathit{\Delta} w_{hi}(t) =& \frac{1}{n_i n_h} S_i(t) \sum_{o \in O} \left[\int_0^\infty a^{pre}(s)[S_o^d(t-s) - S_o^a(t-s)]ds \right] |w_{oh}| \\ &+ \frac{1}{n_i n_h} \sum_{o \in O} \left[S_o^d(t) - S_o^a(t) \right] \left[a + \int_0^\infty a^{post}(s) S_i(t-s) ds \right] |w_{oh}|. \end{aligned} \tag{12}$$

In addition to the STDP weight modifications, synaptic scaling is introduced in order to stabilise the neurons firing activity [14]. Synaptic scaling regulates the strength of synapses in order to keep the neuron's firing rate within a particular range.

The neuron firing rate is kept within an optimal range $[r_{min}, r_{max}]$ by scaling the weights multiplicatively, this way maintaining the relative differences in strength between any inputs [14]. If a weight w_{ij} from neuron j to neuron i causes the postsynaptic neuron to fire with a rate outside the optimal range, the weights are scaled according to the following formula [11]:

$$w_{ij} = \begin{cases} (1+f) w_{ij}, w_{ij} > 0 \\ \frac{1}{1+f} w_{ij}, w_{ij} < 0 \end{cases} \tag{13}$$

where the scaling factor $f > 0$ for $r_i < r_{min}$, and $f < 0$ for $r_i > r_{max}$.

Unlike existing supervised learning rules [12, 6], the multilayer ReSuMe can be applied the networks of spiking neurons with any number of layers. While the tempotron [6] can only be used to discriminate between two classes of spike trains without responding with precise spike patterns, ReSuMe [12] was only used to train single layers or read-out neurons for reservoir networks. On the other hand, SpikeProp [1] can be applied to network with hidden layers, but it is limited to neurons described by Spike Response Model [4] firing single spikes.

The algorithm has been successfully applied to non-linear problems, such as XOR and the *Iris* data set, and to classification tasks on spike train patterns with timescales between 100 and 500 ms [11]. The learning rule has also been used train a spiking neural network with noisy pattern pairs with jitters between 1 and 4 ms. The network correctly classified the patterns above the random performance level where the spikes were moved within a Gaussian distribution with mean 0 and standard deviation up to 10 ms.

3 Experimental Setup

In this paper we are testing a feed forward network of spiking neurons trained with multilayer ReSuMe on noisy and distorted spike train patterns to discriminate highly distorted input patterns.

Three random patterns are fed into the network through 40 input spiking neurons. The hidden layer contains 210 neurons and the patterns are classified by a single output neuron. The input patterns are generated by a pseudo Poisson process with a constant firing rate of $r = 0.1$ within a 500 ms time period, where the spike trains are chosen so that they contain between 15 and 20 spikes (an input pattern has between 600 and 800 spikes). For the spike train generation an inter spike interval is set to 5 ms. In order to ensure that a solution exists, the target patterns are generated as the output of a spiking neural networks initialised with a random set of weights. The target spike trains are chosen so that they contain at least five spikes and no more than seven spikes.

3.1 Network Parameters

The network used for the following simulations is a feed-forward architecture with three layers. The neurons are described by the Spike Response Model [3] (see [11] for details) with the following parameters: the threshold $\vartheta = 0.7$, the time constant of the spike response function $\tau = 7$ ms, the time constant of after-potential kernel $\tau_r = 12$ ms. An absolute refractory period is set for all neurons to $t = 3$ ms. The scaling factor is set to $f = \pm 0.005$. The learning parameters are initialised as follows: $A_+ = 1.2$, $A_- = 0.5$, $\tau_+ = \tau_- = 5$ ms, $a = 0.05$. The weights were initialised with random values uniformly distributed between -0.2 and 0.8. The weights are then normalised by dividing them to the total number of sub-connections.

The learning is considered converged if the network error has reached an average minimum error of 0.5, where the error is defined as the van Rossum distance between the actual spike train and target spike train [13] with the time constant $\tau_c = 10$ ms (see [11] for details). All results are averaged over 20 of trials, with the network being initialised with a new set of random weights every trial. On each testing trial the learning algorithm is applied for a maximum of 2000 iterations or until the network error has reached the minimum value.

4 Results

In [11] it has been shown that a neural network trained on noisy patterns can discriminate input patterns and correctly classify them above the random performance level when jitters of up to 10 ms are introduced. In this paper, a neural network trained on three noisy spike train patterns (each spikes is moved within a Gaussian interval with mean 0 and standard deviation 2 ms) is tested against distorted input patterns. The patterns have been distorted by adding or removing spikes randomly. Figure 1 shows the average accuracy rates (the percentage of input patterns that are correctly classified) for a random set of 150 different distorted spike train patterns, where 10 to 200 spikes have been added or removed. The output of the network is determined as the closest to the target pattern in terms of the van Rossum distance. The network is classifying the patterns where spikes have been added with very high accuracy rates (above 98%). When tested on incomplete patterns, the accuracy rates start to drop as more spikes are removed from the input patterns. Nonetheless, the networks is correctly classifying more than 40 % of patterns (above the random performance level of 33%) even when 200 spikes are missing.

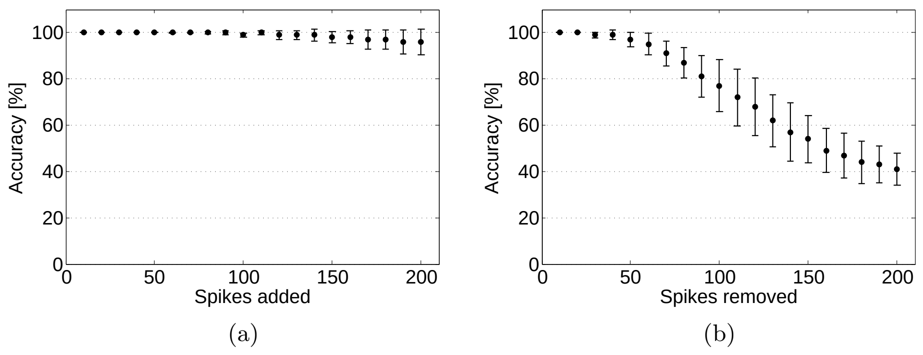

Fig. 1: The average accuracy on distorted patterns for a neural network trained on noisy patterns where each spikes was moved within a Gaussian interval with mean 0 and standard deviation 2 ms: (a) The network is tested against distorted patterns, where 10 to 200 spikes have been added randomly. (b) The network is tested against incomplete patterns, where 10 to 200 spikes have been removed randomly. The error bars show the standard error of the mean.

A feed forward network is trained using multilayer ReSuMe [11] with distorted patterns (10 to 50 spikes are added or removed) and tested against distorted patterns. Although the network is reliably learning the patterns, it needs more iterations to converge as the training patterns have more additional or missing spikes. Figure 2 shows the average number of iterations needed for convergence for networks trained on distorted patterns with 10 to 50 additional or missing spikes.

Figure 3 shows the accuracy rates for a random set of 150 different distorted spike train patterns, where 10 to 200 spikes have been added or removed for a network trained with patterns where 30 spikes have been added. The network is classifying the patterns where spikes have been added with very high accuracy rates (above 98%). Again when tested on incomplete patterns, the accuracy rates start to drop as more spikes are removed from the input patterns. The networks is correctly classifying more than 40 % of patterns when up to 200 spikes are missing. Similar graphs resulted for networks trained with patterns where 10 to 50 spikes have been added or removed.

5 Conclusions

It was shown in [11] that a feed forward spiking neural network with a hidden layer can reliably learn spike timing patterns which range between 100 and 500

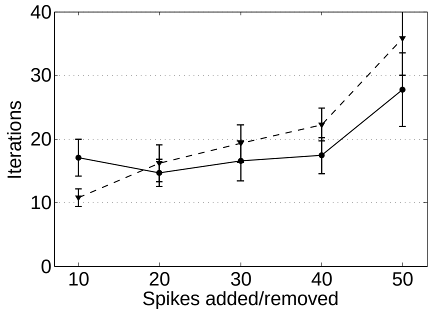

Fig. 2: The average number of iterations needed for convergence for a feed forward spiking neural network trained with distorted input patterns: the circle markers (solid line) represent the average number of iterations when additional spikes were introduced; the triangle markers (dashed line) represent the average number of iterations when spike were removed. The error bars show the standard error of the mean.

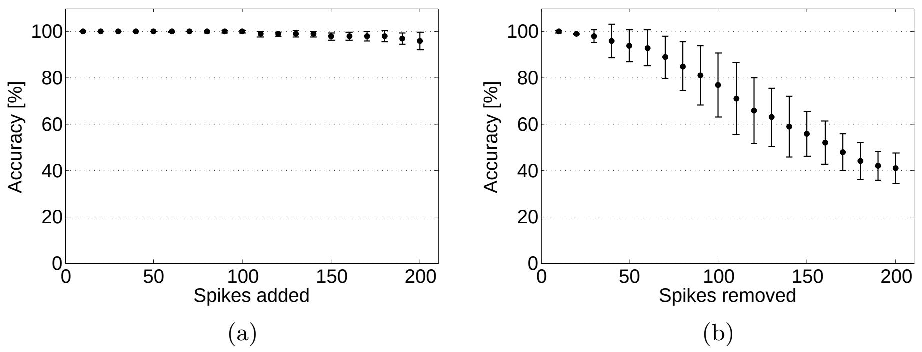

Fig. 3: The accuracy on distorted patterns for a neural network trained on distorted patterns where 30 spikes have been added to the input spike trains: (a) The network is tested against distorted patterns, where 10 to 200 spikes have been added randomly. (b) The network is tested against incomplete patterns, where 10 to 200 spikes have been removed randomly. The error bars show the standard error of the mean.

ms both in noise free environments and when noise was added to the training patterns. In this paper, we examine the network ability to classify spike train patterns with additional or missing spikes trained on noisy and distorted spike timing patterns.

In all training cases, the networks can classify patterns with a large number of additional spikes with very high accuracy rates. The networks also perform well when tested with incomplete patterns, however, the accuracy rates drop as more spikes are missing from the input patterns. The difference in performance between the two sets of simulations can be explained by the absolute refractory period when no spike can be initiated. The absolute refractory period limit the neuron firing rate when the input spike trains have additional spikes. Thus, the network respond with the correct output pattern even when the input patterns have 200 additional spikes. The spiking neural network also correctly classifies highly incomplete patterns above the random performance level.

Acknowledgments. André Grüning was partly supported by EPSRC grant EP/I027831/1.

References

1. Bohte, S., Kok, J., & Poutré, H.L.:Error backpropagation in temporally encoded networks of spiking neurons. Neurocomputing, 48, 17 – 37, (2002)
2. deCharms, R.C., Merzenich, M.M.: Primary cortical representation of sounds by the coordination of action-potential timing. Nature, 381, 610-613 (1996)
3. Gerstner, W.: A Framework for Spiking Neuron Models: The Spike Response Model. In Moss, F., Gielen, S. (eds.) The Handbook of Biological Physics, vol. 4, pages 469–516, Elsevier Science (2001)
4. Gerstner, W., Kistler, W.M.: Spiking Neuron Models. Single Neurons, Populations, Plasticity. Cambridge University Press (2002)
5. Gütig, R., Aharonov, R., Rotter, S., Sompolinsky, H.: Learning Input Correlations through Nonlinear Temporally Asymmetric Hebbian Plasticity. J of Neuroscience, 23(9), 3697 – 3714 (2003)
6. Gütig, R. & Sompolinsky, H.: The tempotron: a neuron that learns spike timing-based decisions. Nature Neuroscience, 9(3), 420 – 428 (2006)
7. Heeger, D.: Poisson Model of Spike Generation. Available online at: www.cns.nyu.edu/ david/handouts/poisson.pdf (2001)
8. Kempter, R., Gerstner, W., Van Hemmen, J.L.: Intrinsic Stabilization of Output Rates by Spike-Based Hebbian Learning. Neural Comp., 13, 2709 – 2741 (2001)
9. Neuenschwander, S., Singer, W.: Long-range synchronization of oscillatory light responses in the cat retina and lateral geniculate nucleus. Nature, 379, 728–733 (1996)
10. Maass, W.: Networks of spiking neurons: the third generation of neural network models. Trans. of the Soc. for Comp. Simulation International, 14(4), 1659–1671 (1997)
11. Sporea, I., Grüning, A.: Supervised Learning in Multilayer Spiking Neural Networks. Under revision, Pre-print: http://arxiv.org/pdf/1202.2249v1 (2012)
12. Ponulak, F., Kasiński, A.: Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting, Neural Comp., 22(2), 467–510 (2010)
13. van Rossum, M.C.: A novel spike distance. Neural Comp., 13(4), 751–63 (2001)
14. Watt, A.J., Desai, N.S. Homeostatic plasticity and STDP: keeping a neurons cool in a fluctuating world. Front. in Synaptic Neuroscience, 2(5) (2010)
15. Wehr, M., Laurent, G.: Odour encoding by temporal sequences of firing in oscillating neural assemblies. Nature, 384, 162–166 (1996)