VALIDATION TESTING OF TEMPORAL NEURAL NETWORKS FOR RBF RECOGNITION

Khaled E. A. Negm
Etisalat College of Engineering
Sharjah, POB 980, UAE

ABSTRACT

A neuron can emit spikes in an irregular time basis and by averaging over a certain time window one would ignore a lot of information. It is known that in the context of fast information processing there is no sufficient time to sample an average firing rate of the spiking neurons. The present work shows that the spiking neurons are capable of computing the radial basis functions by storing the relevant information in the neurons' delays. One of the fundamental findings of the this research also is that when using overlapping receptive fields to encode the data patterns it increases the network's clustering capacity. The clustering algorithm that is discussed here is interesting from computer science and neuroscience point of view as well as from a perspective.

1. Introduction

Artificial neural networks (ANNs) whose functioning is inspired by some fundamental principles of real biological neural networks have proven to be a powerful computing paradigm. Real biological neurons communicate through short pulses, called spikes, which terminate at different time rates. While firing, the firing rate is considered as the relevant information exchanged occasion between neurons, where the analog inputs for an artificial neuron are usually interpreted as firing rates. A spiking neuron is a simplified model of the biological neuron, however it is more realistic than a threshold gate (perceptron) or a sigmoidal gate. One reason for this is that in a network of spiking neurons the input, output and internal representation of information is more closely related to that of a biological network. This representation allows for time to be used as a computational resource and a correlation factor. It has been shown that such a network is computationally more powerful than a network of threshold or sigmoid gates; however learning algorithms for spiking neural networks are still lacking [1].

In order to understand the neural code we have to investigate the temporal structure of the spiking neurons [2-4], where neurobiological findings have confirmed such dependency and have shown that the sign and strength of the change depends on the timing of the two-spike systems [5,6].

The importance of the timing of the first spike has been discussed by many authors in which they were able to show that humans can process visual patterns in 150 ms [7]. Within this time it is hard to imagine that the neurons may sample firing rates, since there are only about 10 synaptic stages involved. Neurons participating in such computations usually have a firing rate of less than 100 Hz hence 10 *ms* were not sufficient to estimate the current firing rate of some spiking neuron [8]. To use a population code, one would need a huge amount of neurons to encode reliably sufficiently many variables.

So far there is no much information known about the possible computational mechanisms on the basis of the timing of single spikes. Some fundamental results have been provided by Maass in which he characterized the computational power of SNNs and showed how the timing of spikes can be used to simulate sigmoidal gates with SNNs [9,10].

On the basis of these principles we show how methods originally designed for artificial neural networks like competitive learning, self-organizing behavior and radial basis functions (RBF) can be realized within this context.

2. Classification of Neurons' Encodes

One can classify how neurons encode information essentially by three different approaches: the first is the *rate coding* where the essential information is encoded in the firing rates and averaged over time or over several repetitions of the event. The second is the *temporal coding*, where the timing of single spikes is used to encode information. And the third is the *population coding*, where information can be distinguished by the activity of different populations of neurons where a neuron may participate at several pools.

In the present study we will focus on the temporal coding in which relevant information could be represented. In this context we consider the firing rates of neurons relative to the stimulus onset such that the closer a neuron fires to the onset the stronger the stimulation can take effect [11]. Hence only the first spike of a neuron carries relevant information. One might also assume that a neuron is shut off by some additional inhibitory input after its firing. Similarly one can assume that information is encoded relative to the firing times of other neurons [12,13]. Which means that we have to inherit a correlation factor between the spiking neurons as considered one of the novel ideas of the current paper.

2.1 Mathematical Model for Temporal Neural

The state of neuron j is described by the variable $h_j(t)$, which models the neuron's membrane potential (Excitatory Post Synaptic Potential-ESPS) at time t. A spike is generated whenever $h_j(t)$ crosses the threshold and $F_j = \{t_j^1, t_j^2, t_j^3, \ldots\}$ is the set of firing times of neuron j. We can model the effect of an incoming spike on the ESPS by

$$\varepsilon(\delta) = \begin{Bmatrix} \frac{\delta}{\tau}\, \boldsymbol{exp}\left(1 - \frac{\delta}{\tau}\right), & \forall\ \delta > 0 \\ 0, & \forall\ \delta < 0 \end{Bmatrix} \quad (1)$$

where δ is the post time to the spike effect and τ is the time to the peak of the post synaptic reaction.

When the neuron j receives an input from a set of neurons marked as Γ_j, the membrane potential becomes the weighted sum of EPSP's caused by the neurons from Γ_j. In this case

$$h_j(t) = \sum_{j \in \Gamma_i} \sum_{t_j^k \in F_j} w_{ij} \varepsilon(t - (t_j^k + d_{ij})) \quad (2)$$

where w_{ij} is the synaptic efficiency (i.e., weights) between neurons i and j, d_{ij} is the delay from the occurrence of a spike in neuron j and the beginning of its effect on neuron i and $(t_j^k + d_{ij})$ is the time when the k^{th} spike from neuron j started affecting neuron i.

Throughout the present work we stress on the fact that the synaptic efficiency is the only learnable parameter that is of interest to us to investigate.

2.2 NNS Architecture and Data Temporal Encoding

Assume that m-dimensional input patterns $(x_1,\ldots,x_m)$ are encoded with sensory neurons $u_1,\ldots,u_m$ which are the constituents of the network inputs of an n-dimensional output neurons $(v_1,\ldots,v_m)$ (see Figure 1). In the simplest case each input neuron u_i forms one synaptic connection to each RBF neuron v_j with weight w_{ij} and delay d_{ij}.

In the present study we consider a coding scheme where each input neuron u_i fires exactly once at time t_i during the coding interval $[t_i, (\delta_t + t_i)]$, i.e., with δ_t is a constant time step. The input is encoded relative to the first spike in the coding interval so that explicit reference spike is not necessary. We will denote an input vector for this type of coding with $\boldsymbol{x}$ that is defined by the vector

$$\boldsymbol{x} = \langle x_1, \ldots, x_m \rangle \quad (3)$$

where

$$x_i = \boldsymbol{max}\left\{ t_i \mid 1 \le i \le m \right\} - t_i \quad (4).$$

We can define the center of an RBF neuron (which is symmetric around the centre) c_j by the vector

$$\boldsymbol{c}_j = \langle c_j, \ldots, c_{j,m} \rangle \quad (5)$$

where

$$c_{j,m} = d_{im} - \min\left\{ d_{im} \mid 1 \le i \le m \right\} \quad (6)$$

Assuming that the RBF neuron v_j is associated with a m-dimensional vector $\mu_j^c = (\mu_j^1, \mu_j^2, \ldots, \mu_j^m)$. In that sum we define

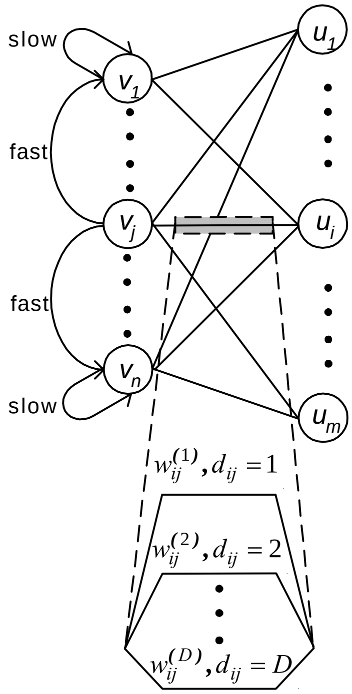

Figure 1. The SNN Encoding architecture

the m-dimensional vector as the synaptic weights of the neuron. Each coordinate in μ_j^c is a weighted average of the delays from the matching input coordinate, i.e.,

$$\mu_j^c = \sum w_{ij}^{(d)} \quad (8)$$

The active synapse from μ_j will be the one whose delay is μ_j^c as

$$w_{ij}^d = \begin{cases} w_{\max}, & d = \mu_j^i \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

The center of such a neuron is mirror image of μ_j^c and the response time to input patterns is symmetric around that center so that this neuron realizes a RBF. For example, if $\mu_j^c = (1,4,7)$, v_j's center be (6,3,0).

In Figure 1, each connection from a sensory neuron to an RBF neuron consists of D synapses with delays *1,..., D* (the marked gray area between u_i and v_j). Each sensory neuron emits exactly one spike when the pattern is exposed to the network. The spike from u_j is delayed by x_j in milliseconds. For example, the 3-dimensional input (3,7,1) will be coded by u_1 firing after *3 ms*, u_2 firing after 7 *ms* and u_3 firing after 1 *ms*. Inhibiting synapses between all of the activated RBF neurons when they fire will implement a winner-take-all-mechanism to allow only one RBF neuron to respond to each pattern.

For each pair of sensory neuron u_i that is connected to a single layer of n spiking neurons and RBF neuron v_j, there exist a set of D independent synapses. Taking into account the multiple synapses and constant delays, the weights of these synapses are $w_{ij}^{(1)}, w_{ij}^{(2)}, \ldots w_{ij}^{(D)}$ and the delays are 1 *ms*, 2 *ms*, ... and *D ms,* respectively. In this case $h_j(t)$ is redefined as

$$h_j(t) = \sum_{j \in \Gamma_j} \sum_{t_j^k \in F_j} \sum_{d=1}^{D} w_{ij}^{(d)} \varepsilon(t - (t_j^k + d)) \quad (7)$$

The input $\boldsymbol{x}$ is close to the center c_j of an RBF neuron v_j if the spikes of the input neurons reach the soma of v_j due to the corresponding delays at similar times, i.e., if $\| \boldsymbol{x} - \boldsymbol{c}_j \|$ is small. This is basically a parallel approach to that was introduced by Hopfield in 1995 in which he considered the case where the input vector is close enough to the center of an RBF neuron to make v_j fire [14].

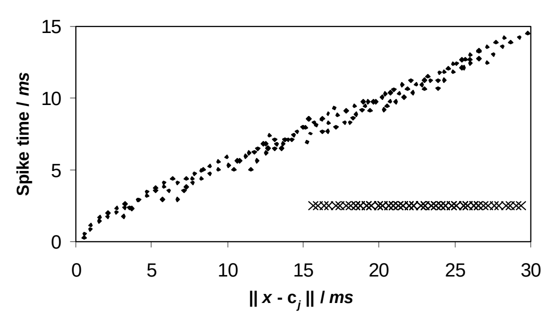

Figure 2: Dependence of the firing time of an RBF neuron v_j

If the distance between $\boldsymbol{x}$ and v_j is too large, v_j does not fire at all. If for some input vector $\boldsymbol{x}$ the difference $\| \boldsymbol{x} - \boldsymbol{c}_j \|$ is small enough for various j to make v_j to fire then the RBF neuron whose center is closest to x fires first. In this case a set of such RBF neurons can be used to separate inputs into various clusters.

Figure 2, shows the dependence of the firing time of an RBF neuron v_j on the distance of the input vector $\boldsymbol{x}$ to the center c_j. For this simulation 1200 uniformly distributed inputs $\boldsymbol{x} \in [0, 30ms]^{1200}$ are presented to v_j with equal weights and delays uniformly distributed over $[0, 30ms]$. Crosses indicate the case that the RBF neuron has not fired. (RBF++ Reference)

The delays between spikes in such an input pattern will be evened out by the delays in the synapses causing v_j to react to the all incoming spikes at the same time. When comparing the reaction of v_j to an input pattern which is at its center and an input pattern which is slightly off its center we find that the first pattern causes a higher peak in v_j's membrane potential and the membrane potential crosses the threshold earlier.

3. Learning RBF and Clustering with Temporal Neurons

The learning goal of the network is to have one RBF neuron related to each cluster so that when an input pattern from that cluster is exposed to the network, only the related neuron will fire. Since we have inhibitory synapses between the neurons, it is enough that the correct neuron will fire first. In order to achieve that, the weights of the spiking neurons will be shifted during learning so that they will be able to realize RBFs whose centers are the centers of the clusters.

The learning rule (which is a variant of the Hebb law) is applied to the synaptic weights of the neuron that is fired when the input is fed to the network. Synapses which contributed to the neuron's firing are strengthened and synapses which did not contribute are weakened. The synapses that contributed are those who started affecting the postsynaptic neuron (taking into account the synaptic delay) slightly before the neuron actually crossed the threshold. The change in the synaptic weight is given by the following learning rule

$$\Delta w_{ij}^{(d\ )} = \eta\ L((t_i - d\ \ ) - t_j) \quad (10)$$

where η determines the learning rate.

The learning function will have the form

$$\alpha(\delta_t) = (1-b)\,\boldsymbol{exp}\left(-\frac{(\delta_t - c)^2}{\beta^2} + b\right) \quad (11)$$

where b is the minimal value, c is the location of the peak, β is the width of the distribution.

We found out that with time difference of exactly c *ms* between the spikes causes maximal strengthening of the synaptic weight up to the width of the distribution (higher or lower) causes a smaller strengthening and weakens the synaptic weight. This is a very good result that guided us to the best resonance value for the system parameters. With the use of this above model data is successfully clustered with results similar to [15].

4. Input Encoding with Receptive Fields

In order to encode the sensory input neurons we should look for an efficient encoding technique. Sensory input in live organisms are often encoded with overlapping receptive fields, for example touching the skin at a certain area may cause several sensory neurons to fire at different rates. This technique is used to encode input patterns so that it is possible to successfully cluster more complicated data sets.

When using receptive fields, input is not encoded by using just one sensory neuron for each data coordinate. Instead for each coordinate, several receptive field neurons are used to encode the data. Each of the receptive fields fires with a short delay if the value of that coordinate is close to its center and with a longer delay for values farther from the center. And it does not fire at all if the value is too far from the center. The centers of the receptive fields are evenly distributed within the possible range of values.

When a pattern is introduced to the system each receptive field calculates the value of a gaussian function and value of the input at the coordinate will correlate with each other. The gaussian function for each receptive field is given by

$$\boldsymbol{exp}\left(-\frac{(x_i - c_j)^2}{\gamma^2}\right) \quad (12)$$

where γ is the width of the receptive field

$$\gamma = \beta_{rf}\ M/(y\text{-}2)\ (13)$$

with M being the maximal value of the input, y the number of receptive fields and β_{rf} a parameter.

5. Model Implementation

The model was implemented by C++ using the three classes ReceptiveFiled, RBFneuron and Network Parameters' Monitoring. The algorithm contains main three modules. The first one is data set file creation module, the second is data set file testing module and the third one is the distance from the centre, response time testing and running module.

The results can appear in a different folder in an ascii format and the weights of the RBF neurons before and after learning are recorded in a different ascii file.

Before running the algorithm code some pre-set definitions and values for key parameters should be assigned. Those values are:

- *Maximal weight value,* The maximal weight is calculated so that after learning the active synapses that are left have enough weight to cause the RBF neurons to fire. There is also a minimal weight which is set to zero in which no inhibitory synapses are allowed.
- *Saturation function* that is needed in order to keep the weights within the realistic range $[0, w_{max}]$. This function causes the synaptic weight to change at a slower rate when close to 0 or γ_{max}.
- *Initial weight value* where the weights are initiated randomly but they must be high enough to cause some neuron to fire for every pattern and low enough so that all the input spikes are necessary in order to cause a RBF neuron to fire. If not all input spike are necessary a partial pattern will be learned and several clusters with a common sub-pattern will be identified by one of the neurons.

6. Results and Conclusion

Throughout this work we incorporated receptive fields along with the distance from the center versus the firing time function of the RBF neurons under test. A single spiking neuron with one active synapse from each sensory neuron is stimulated with different input patterns and the time of firing is measured and compared to the distance of the patterns from the neuron's center.

Figure 3 shows the results of this run, in which the spike time is measured relative to the minimal response time. Spike time "-1" indicates that the neuron did not fire as a result of that pattern.

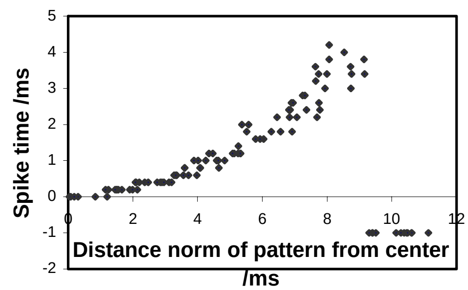

Figure 3: firing time of the RBF neurons versus distance of the input patterns from the RBF center.

The value of the gaussian (between 0 and 1) is normalized to a value less than 1 and the result is multiplied by the maximal delay to the delay of the receptive field to that input. If the value of the gaussian is too low no spike will be emitted by that receptive field. In Figure 4, The vertical line indicates an input which causes three receptive fields to fire, the central receptive field will fire almost immediately and the ones to its left and right will fire after a longer delay of value 80% and 90% of the maximal delay.

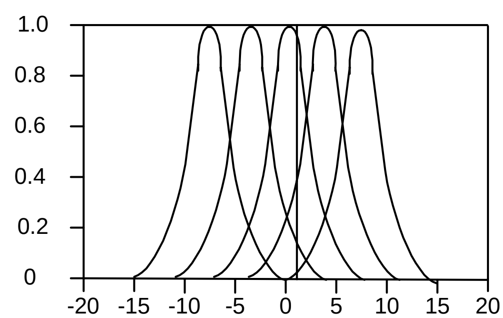

Figure 4. The Gaussian receptive fields

In table 1 we shows the results for different data sets. We run the algorithm for different values of dimensions for the data sets. It shows very promising results in terms of the standard deviation relative to the number of data points.

In Figure 5 we shows the relation for the first row of the table, and it is obvious that it gives a very good results to discriminate and cluster data points in a very small response time.

Table 1: Data Set Parameters used for NNS Model Testing

Dimension	# of Clusters	Range *ms*	Std *ms*	# of patterns
2	20	**0 - 30**	0.2	2600
10	20	**0 - 30**	0.22	3000
20	20	**0 - 30**	0.23	3500
40	20	**0 - 30**	0.24	2800
60	20	**0 - 30**	0.26	4000
80	20	**0 - 30**	0.28	4000
4-Iris data	12			1000

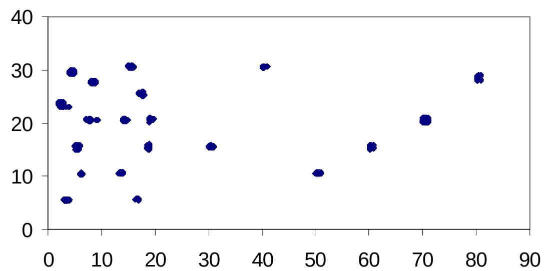

Figure 5: Plot of the 2-dimensional data set results for the NNS testing.

In all the simulations 30% of the data was used for learning. Testing was performed on the full data sets. In the artificial data sets 100% correct classification was achieved. In the iris data set over 95% success was achieved.

So we can see that spiking neurons, receiving temporally encoded inputs can compute radial basis function to an excellent accuracy. This is feasible via sorting the relevant information in their delays. In the current study we showed how our models introduced excellent results with more simpler buildup than the previous studies.

7. Future Work

Currently we are studying applying this technique to more NNS application oriented problems. Of main interest to us to benefit from the short time convergence into correct clustering and very small standard deviation. Namely applying this for intrusion detection systems as extensions to our previous efforts in that field [16,17].

8. References

[1] MAASS, W., Networks of spiking neurons: the third generation of neural network models, Neural Networks, 10:1659-1671, 1998.

[2] SEJNOWSKI, T., Time for a new neural code? Nature, 376:21-22, 1995.

[3] ABELES, M., CORTICONICS, A., Neural Circuits of the Cerebral Cortex, Cambridge University Press, 1991.

[4] ABELES, M., Bergman, H., MARGALIT, E., and VAADIA, E., Spationtemporal firing patterns in the frontal cortex of behaving monkeys, Journal of Neurophysiology, 70(4):1629-1638, 1993.

[5] MAINEN, Z. F. and SEJNOWSKI, T., Reliability of spike timing in neocortical neurons. Science, 268:1503-1506, 1995.

[6] RIEKE, F., WARLAND, D., BIALEK, W., and de RUYTER VAN STEVENINCK, R., SPIKES: Exploring the Neural Code, MIT-Press, Cambridge, 1996.

[7] THORPE, S., FIZE, D., and MARLOT, C., Speed of processing in the human visual system. Nature, 381:520-522, 1996.

[8] THORPE, S. and IMBERT, M., Biological constraints on connectionist modeling. In Pfeifer, R., Schreter, Z., Fogelman-Soulie, F., and Steels, L., editors, Connectionism in Perspective. Elsevier, 1989.

[9] MAASS, W., Lower bounds for the computational power of networks of spiking neurons. Neural Computation, 8:1-40, 1996.

[10] MAASS, W., Fast sigmoidal networks via spiking neurons, Neural Computation, 9:279-304, 1997.

[11] GAWNE, T. KJAER, T. and RICHMOND, B., Latency: Another potential code for feature binding in striate cortex, Journal of Neurophysiology, 76:1356-1360, 1996.

[12] O'KEEFE, J. and REECE M., Phase relationship between hippocampal place units and the EEG theta rhythm. Hippocampus, 3:317–330, 1993.

[13] THORPE, S. and IMBERT, M., Rapid visual processing using spike synchrony. In Advances in neural processing systems, vol. 9, 901-907. MIT Press, Cambridge, 1996.

[14] HOPFIELD, J., Pattern recognition computation using action potential timing

for stimulus representation, Nature, 376:33-36, 1995.

[15] NATSCHLÄGER, T. and RUF, B., Spatial and temporal pattern analysis via spiking neurons Network, Computation in Neural Systems, 9:319-332, 1998.

[16] NEGM, K. "Real Time UNIX Kernel Based Intrusion Detection System", Proc. of International Conference of Intelligent Systems and Control-Oct. 2002, Japan, pp 126-138.

[17] NEGM, K., "Double Step Intrusion Detection System", Proc. of International Conference of Intelligent Systems and Control-June 2003, Austria, pp 26-32.