2014 International Conference on Advances in ICT for Emerging Regions (ICTer): 074 - 077

An Efficient Discrete Model for Implementing Temporal Coding Spiking Neural Network

E Y Andrew Charles

Department of Computer Science, University of Jaffna
Thirunelveli, Sri Lanka
charles.ey@jfn.ac.lk

Abstract—**Spiking neurons, which makes up SNNs are more biologically realistic artificial neurons which convey information by the timing of spikes. SNNs have been applied for various learning tasks and they are capable to perform with improved accuracy in less number of training cycles. Due to the temporal nature a spiking neuron cannot be implemented as a standard neuron. Generally spiking neurons are implemented as a discrete model and the output is computed through an iterative process. Here, each iteration is assumed to take one unit time. Even though the learning models of SNNs can learn in less number of cycles, they consume more computational resources due to the large number of iterations they require. This paper proposes a numerical approach using Newton-Raphson method to estimate the firing time of a spiking neuron in lesser number of iterations.**

Proposed model was implemented and analysed to test its accuracy, convergence and capability for learning. Tests were performed using the Iris plants dataset and Breast cancer dataset from UCI Machine learning repository. The proposed model was able to find the actual neuron firing time within only a few iterations. Tests have shown that for the proposed approach the required iterations needed to estimate the firing times to an accuracy of two decimal places would be at least 400 fold lesser compared to the standard approach. Number of required iterations found to be independent of the firing time in contrast to the standard approach. The model also found to be smoothly converges in finding the actual firing time. Further the model was tested to cluster Iris and Cancer data sets and the accuracy was found to be 92.7% and 97.4 % respectively. The results demonstrate that the limitation of estimating the actual firing time with less computational resources can be resolved using the proposed discrete approach. This would pave way to devise better learning algorithms for SNNs with even higher accuracy and low training cycles.

Keywords— **Spiking Neural Networks, Temporal coding, Numerical approximation.**

I. INTRODUCTION

Artificial Neural Networks (ANNs) are a powerful and flexible computational tool for solving a wide range of problems in many domains. Although the creation and development of ANNs were inspired by biological neural systems, ANNs are considered to be limited compared to their biological counterpart due to their simplistic structure and behaviour [1, 2]. These considerations have led to increased interest in temporal coding spiking neurons which are more biologically realistic artificial neurons and in Spiking Neural Networks (SNNs). Generally, in biological neural systems, information is conveyed to neurons through neural connections by electric pulses called spikes. It was widely believed that the information is coded in the frequency of the spikes [3]. The analogue variable in classical neural networks corresponds to the firing rate of a neuron [2]. However, experimental evidence collected in recent years indicates that many biological neural systems use the timing of single spikes to encode and process information [2, 4]. This method known as temporal coding is considered as the coding mechanism in biological neural systems for fast cortical events [4].

Networks of spiking neurons with temporal coding have become an important research area. Hopfield [5] introduced the idea of using the timing of action potentials to represent the values for computation within a network. Maass [2] showed that a network of spiking neurons can simulate arbitrary feed-forward sigmoidal networks (ANNs with sigmoidal activation) and can approximate any continuous functions. It has been proved that spiking neural networks which convey information by individual spike times are more computationally expressive than networks with sigmoidal activation [6]. Another feature of spiking neurons is that even with a seemingly increased structural complexity they are relatively easier to implement in large neural networks [7]. Different learning algorithms for SNNs with temporal coding were proposed in [7-13] and their efficiency was found to be comparable with that of popular sigmoidal neural network models.

Temporal coding SNNs have demonstrated their capability in various learning applications [13-14]. Even though these applications were shown to produce better results in fewer training iterations, SNNs consumed more computing resources due to the way they are implemented in a digital computer. Temporal coding SNN is a continuous model and the input and output of a SNN are spike times. A digital computer can implement only a discrete model, and the continuous nature is approximated by iterating the discrete model repeatedly considering each iteration takes a unit time interval. To produce results equal to the continuous model, these discrete models need to be iterated for a large number of time, consuming great amount of resources, namely CPU time and memory. For example to find the firing time of a spiking neuron with an accuracy of two decimal places, computation needed to be iterated for more than 100 fold of the firing time of the neuron considering time interval as 0.01 between each iteration. On the other hand sigmoidal neural networks can produce their output with high accuracy for a particular input in a single computation. In-order to limit the need of computational resources of SNNs, number of iterations is often reduced leading to low accuracy of the output.

This research proposes a novel method to implement the SNN with less computational resources but with high accuracy. Proposed model iteratively computes the actual firing time of a spiking neuron starting from an approximate firing time. Here the computation of the firing time is

978-1-4799-7732-1/14/$31.00 ©2014 IEEE

performed using a numerical approach, namely the Newton-Raphson method for finding roots of an equation. The proposed method is implemented on a Self Organising Spiking Neural Network and tested on standard data sets for its accuracy, convergence and capability for learning.

II. SPIKING NEURAL NETWORK

Spiking neuron models are simple phenomenological models which describe the biophysical mechanisms responsible for generating the activity of a neuron by means of its membrane potential. SNN is a network of spiking neurons where the neurons are usually placed in layers and connected through weighted connections. In addition to a weight, a connection in a spiking neural network could have a delay mechanism which postpones the arrival of an input spike at the other end of the connection.

This study is based on the spiking neural network model introduced in [15]. The model employed in this study can be defined as follows:

Let Γ_j be the set of neurons presynaptic to neuron j and Π the set of firing times of the spikes from all the neurons $i \epsilon \Gamma_j$. The state of neuron j at time t is specified by its potential which can be computed by (1).

$$U_j(t) = \sum_{i \epsilon \Gamma_j} w_{ji} \varepsilon \left(t - \left(t_i + d_{ji}\right)\right), \left[t_i \epsilon \Pi : t_i + d_{ji} < t\right] \quad (1)$$

$$\varepsilon(t) = \frac{t}{t_c} e^{\left(1 - \frac{t}{t_c}\right)} \quad (2)$$

where w_{ji} is the connection weight and d_{ji} is the connection delay between neurons i and j. $\varepsilon(t)$ is the spike response function which specifies the effect of an input spike at some time t. A typical spike response function is given by (2) and is shown in Fig. 1. t_c is a constant called time constant which influences the peak in the spike response function.

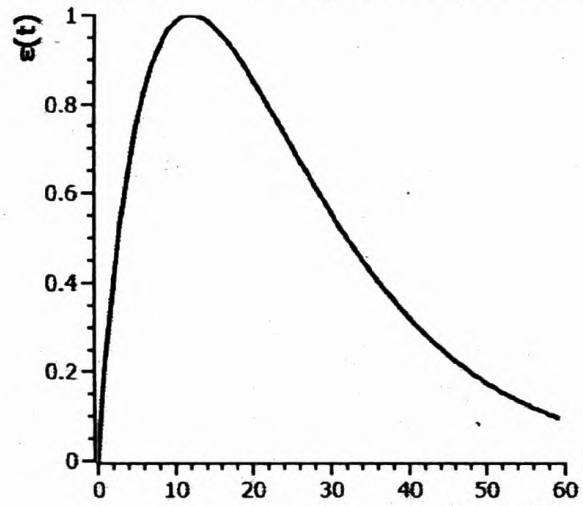

Fig. 1: Spike response function, here the time constant τ=12.

The functioning of a spiking neuron is time related. To compute with an SNN, it has to be activated over a period of time. During this time, the inputs are given at specified times as spikes. Output spikes are generated depending on the timing of the inputs, network connections and the parameters of the neuron. The output of the network is the timing of these spikes. Since input to the network is the timing of input spikes, the continuous analog input values are coded using small temporal differences [11]. For realising the temporal coding a high input value is represented by an early input spike and a low value by a late spike with reference to an input time window [16].

III. A DISCRETE APPROACH FOR IMPLEMENTATION OF SNN

The difference between biological systems and digital systems is that the signals in the former are continuous and in the latter are discrete. Hence, a suitable model is necessary to realise the continuous functionality of an SNN on a digital computer. A common approach to simulate an SNN, the continuous time window is divided into a number of discrete time intervals. Each time interval is considered as a step in the computation. For each step the state of the network is updated [17].

A neuron fires only when its potential due to the incoming spikes exceeds the threshold value. Due to the non-linear nature of the connections of a SNN, the effect of an input spike will increase with time to reach a maximum value and will dissipate over time as shown in Fig. 1. The total potential of a neuron will increase with each input but at the same time it may decrease due to the nature of the input spike. Fig. 2 shows the fluctuation of the neuron potential in a spiking neuron with multiple inputs at different times.

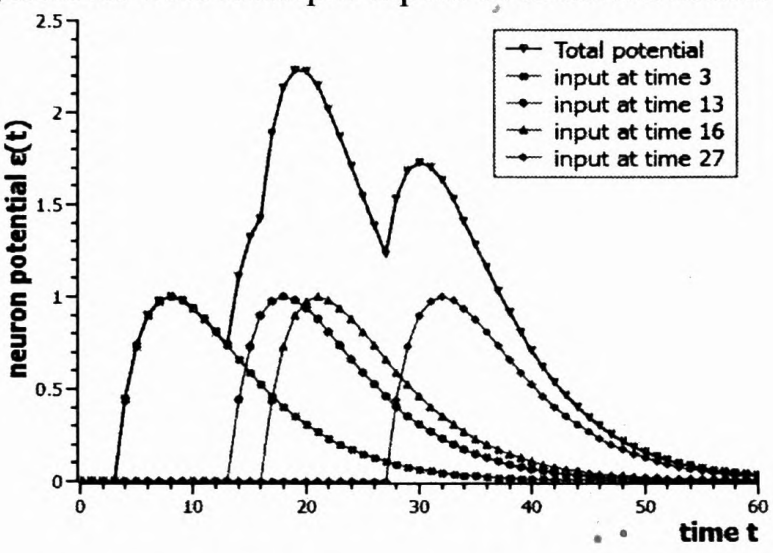

Fig. 2: Change in total neuron potential due to inputs at different times (t_c=5). The output of the neuron is the time when the neuron potential exceeds the threshold value.

A. *Computing the firing time*

Suppose there are n inputs at times $t_1, t_2, t_3, \cdots, t_i, \cdots, t_n$

neuron potential at time T due to spike at time t_i can be specified by (3) using (2):

$$u_j = w_{ji} \left\{ \frac{\left(t - \left(t_i + d_{ji}\right)\right) \times exp\left[1 - \frac{t - \left(t_i + d_{ji}\right)}{t_c}\right]}{t_c} \right\} \quad (3)$$

Suppose the neuron fires at time T, then at the time of firing the total potential of neuron j can be computed by (4) using (1). At this time the neuron potential would be equal to the threshold value θ.

$$U_j(T) = \sum_{t_i} w_{ji} \left\{ \frac{\left(t - \left(t_i + d_{ji}\right)\right) \times exp\left[1 - \frac{t - \left(t_i + d_{ji}\right)}{t_c}\right]}{t_c} \right\} = \theta \quad (4)$$

Let $f(t) = U_j(t) - \theta$. Applying the Newton's method a near exact firing time T′ can be computed using (5) with an approximate estimate T_0'.

$$T_n' = T_{n-1}' - \frac{f(T_{n-1}')}{f'(T_{n-1}')} \quad (5)$$

B. *Estimating the approximate firing time*

An estimate to the firing time can be computed utilizing the coincidence detecting feature of the spiking neuron. A spiking neuron reaches its highest neuron potential at a time from which the sum of differences to the input firing times is minimal. If the effective input times, i.e., the time the effect of an input is realized at a neuron after passing through the connection delay, are ordered the spike at the midpoint would have the minimal sum of differences with other inputs.

Since the connection delays are random values over a specified range the effective inputs will be in the range of input range and delay range added together. Midpoint of this range is a good approximation rather than ordering the inputs and finding the mid. The actual firing time can be computed with Newton's method using this approximate initial value observation and the threshold value.

C. Implementation

The proposed approach is implemented as a temporal coding spiking neural network on a delay adapting self organised learning spiking neural network. The learning model for this study is based on the delay shifting rule proposed in [17] and analysed in [13, 14]. The rule to update the delay of a connection from neuron i to j is specified in (6).

$$\delta d_{ji} = \eta g(\delta t) d(j, winner) \quad (6)$$

$$g(\delta t) = \delta t \left(\frac{e^{-\delta t^2/\tau_{stdp}^2}}{\tau_{stdp}} \right) - b \quad (7)$$

In (6) η is the learning rate and function $g(\delta t)$ is specified in (7) is the amount of change due to the time difference δt. τ_{stdp} is the time constant for the learning rule; b is a positive bias value. δt is the difference between the time of arrival of an input spike and the time of an output spike which can be computed by (8).

$$\delta t = T_j' - (t_i + d_{ji}) - s \quad (8)$$

where t_i is the firing time of input neuron i and T_j' is the firing time of spiking neuron j. d_{ji} is the delay of the connection between neurons i and j. s is a positive term the effect of which is to shift the origin of the time axis in the positive direction. Both b and s are control parameters used in the learning process.

Self-organisation in the proposed model is achieved through competition and cooperation of the spiking neurons [19]. The competition is created based on the firing times of the output neurons. A highly activated neuron will tend to fire earlier [3]. The neuron that fires first (the winning neuron) will have a high degree of coincidence among its inputs, thus having the most accurate delay pattern to compensate for differences in input timings. Therefore the winning neuron is more likely to represent the input vector [7, 10, 11]. Cooperation among the neurons is achieved through lateral excitation. Neurons which are closer to the winner neuron are given precedence over the other neurons. Lateral excitation is realised through physical location of a neuron relative to the winning neuron like in the Kohonen's SOM [19]. During the training process connections of the neurons which are laterally closer to the winner neuron are updated proportionally to its distance from the winner neuron. The effect on neuron i relative to neuron j is given by (9).

$$d(l,m) = e^{-distance(l,m)^2 / 2\sigma^2} \quad (9)$$

where *distance(i, j)* is the lateral distance between neurons i and j in the output lattice; σ is the maximum width of the lateral excitation.

IV. Results

The proposed method was implemented on a Self Organising Spiking neural network and analysed using the Iris plants dataset and Breast Cancer dataset [18], two of the best known datasets to be found in the pattern recognition literature. The Iris dataset contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other. Attribute Information of the dataset are sepal length (in cm), sepal width (in cm), petal length (in cm), petal width (in cm) and class (Iris Setosa, Iris Versicolour, Iris Virginica). Cancer dataset consists of 683 records each with 9 attributes and representing 2 classes namely, Benin and Malignant.

Table I shows the network output for a sample input data (7.0, 3.2, 4.7, 1.4) from the Iris dataset which is represented by spike time code (2.5014, 7.9654, 4.6393, 8.3331). Connection delays for the spiking neuron considered were set as (1.5957, 0.1054, 3.898, 3.7423). Here time constant t_c was set as 20 and threshold of a neuron was set as 3.60 (4 x 0.90). For the simulation input times were in the range [0:10) and the connection delays were in the range [0:5). In order to study the effects of the delays, weights of all the connections were set as 1.0. Table I list the output of the proposed model and the standard model in computing the firing time.

Fig 3 shows the estimation of firing time of the standard model for the selected sample. For the standard model each iteration was done with one unit time interval. At iteration 21 (t=21.00) the neuron potential is less than the threshold value and at iteration 22 (t=22.00) it exceeds the threshold, hence the neuron spike occurs between these two points. Note that it is not possible to find the exact firing time of the neuron with this setup unless the time interval between each iteration is kept at least as low as 0.01. For the above mentioned example it would require at least 2200 iterations to find the firing time with accuracy of two decimal places. In the proposed method spike time can be accurately estimated as shown in Table I. Here the neuron potential reaches the threshold value at time 21.0731 at the fourth iteration. Here the accuracy is much higher (the spike time is computed using a variable of type double in Java) but only five decimal places of the spike time values are shown in the table.

TABLE I Output of a Neuron

Proposed Model		Standard Model	
Iteration	**Spike Time estimate**	**Iteration**	**Neuron Potential**
0	10.00000	17	2.9731
1	18.60199	18	3.1644
2	20.64493	19	3.3300
3	21.05719	20	3.4719
4	**21.07312**	21	**3.5920**
5	21.07312	22	**3.6922**
6	21.07312	23	3.7740

It was also observed that the number of iterations required for proposed model to estimate the firing time is independent of the firing time unlike the standard model, where the firing time is actually the number of consecutive iterations required for the neuron to reach the threshold. Table II lists the average required iterations for the proposed method for various threshold values. A slight increase is observed in the number of required iterations with the increase of threshold value.

Convergence of the model is also analysed. Fig 4 shows the estimated values for the firing time of the neuron at each

iteration of the proposed model for the sample input mentioned above.

TABLE II ITERATIONS REQUIRED TO COMPUTE A SAMPLE OUTPUT

Threshold	Output range	Iterations (average+/-stdev)
Cancer dataset (Input + Delay range: 0-15, time constant: 20)		
9 x 0.80	11.75 - 48.44	4.15+/-0.40
9 x 0.85	13.53 - 46.70	4.82+/-0.42
9 x 0.90	15.37 - 43.02	4.95+/-0.29
9 x 0.95	17.49 - 39.45	5.72+/-0.49
Iris dataset (Input + Delay range: 0-15, time constant: 20)		
9 x 0.80	12.86 - 19.04	4.01+/-0.12
9 x 0.85	13.72 - 19.69	4.32+/-0.47
9 x 0.90	14.89 - 21.69	5.00+/-0.00
9 x 0.95	17.92 - 23.68	5.37+/-0.49

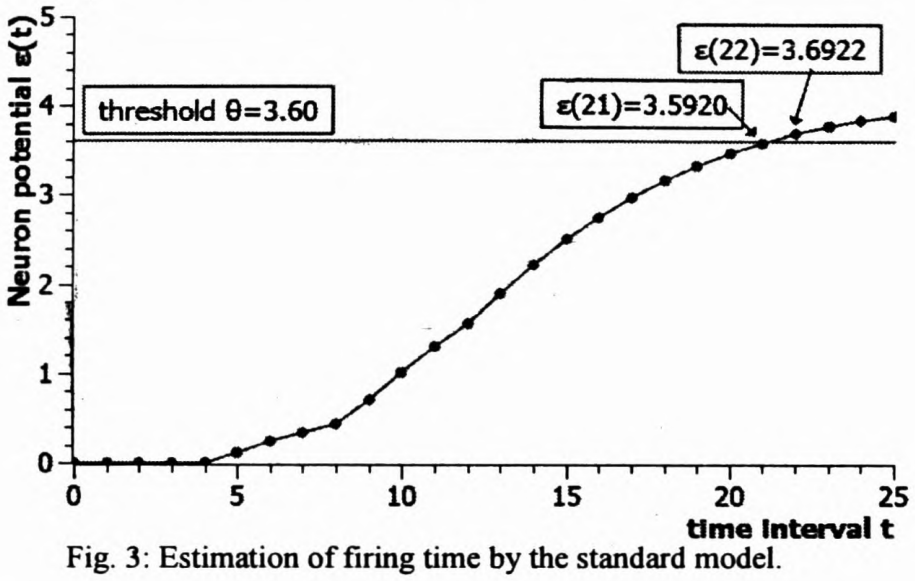

Fig. 3: Estimation of firing time by the standard model.

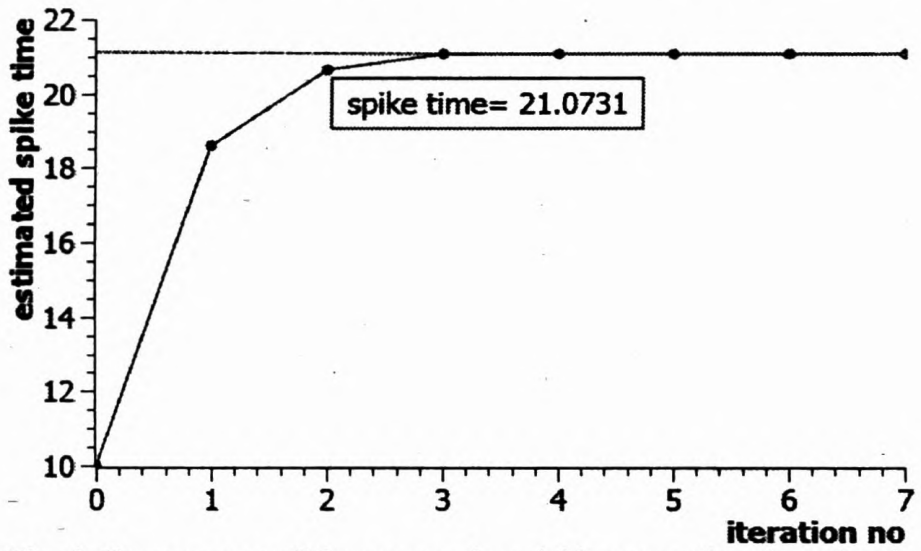

Fig. 4: Convergence of the proposed model in computing the spike time.

To analyse the accuracy of the computed spike times, the model was applied to cluster the Iris and Cancer datasets. Model was applied to cluster different order of the datasets for 5 times. Average clustering accuracy was found to be 92.7% for the Iris dataset and 97.4% for the Cancer dataset. Cluster formation is made with the positions of the winner neurons and their firing times. Results shows that the network preserves the topology of the input data accurately. It was observed that same set of winner neurons were firing within different time ranges for different classes. This observation was utilised to overcome the problem of overlapping winner neurons.

V. CONCLUSIONS

In this paper, a discrete model for implementing spiking neural network is proposed and analysed. The efficiency of the model has been demonstrated by applying it to compute the firing times and cluster benchmark datasets. The proposed model was able to estimate the accurate firing times of the neurons with only a few iterations. The classification accuracy obtained for the proposed model is comparable to that of accuracy reported for Sigmoidal and Spiking Neural Network learning models[13] but with only a fraction of computational effort. The results presented in this paper demonstrate that the limitation of estimating the actual firing time can be resolved using the proposed discrete approach and better learning algorithms can be devised for spiking neural networks.

ACKNOWLEDGMENT

This work is a continuation of research done at the Manufacturing Engineering Centre, School of Engineering, Cardiff University, Cardiff, UK. under the supervision of Prof. D. T. Pham and Dr. M. S. Packianathar.

REFERENCES

[1] A.M. Zador, "The basic unit of computation," Nature Neuroscience supplement, 3, 2000, p. 1167.

[2] W. Maass, "Fast sigmoidal networks via spiking neurons," Neural Computation," 9(2), 1999, pp. 279-304.

[3] S.J. Thorpe, A. Delorme and R. van Rullen, "Spike-based strategies for rapid processing," Neural Networks, 14(6-7), 2001, pp. 715–726.

[4] W. Gerstner, R. Kempter, J.L. van Hemmen and H. Wagner, "Neuronal learning rule for sub-millisecond temporal coding," Nature, 383(6595), 1996, pp. 76-81.

[5] J.J. Hopfield, "Pattern recognition computation using action potential timing for stimulus representation," Nature, 376, 1995, pp. 33–36.

[6] W. Maass, "Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons," in Advances in Neural Information Processing Systems, volume 9, MIT Press, Cambridge, 1997, pp. 211-217.

[7] S.M. Bohte, J.N. Kok and H. La Poutre, "Error-back propagation in temporally encoded networks of spiking neurons," Neuro computing, 48, 2002, pp. 17-37.

[8] B. Ruf and M. Schmitt, "Learning temporally encoded patterns in networks of spiking neurons," Neural Processing Letters, 5(1), 1997, pp. 9–18.

[9] B. Ruf and M. Schmitt, "Self-organisation of spiking neurons using action potential timing," IEEE Transactions on Neural Networks, 9(3), 1998, pp. 575–578.

[10] T. Natschläger and B. Ruf, "Spatial and temporal pattern analysis via spiking neurons," Network: Computational Neural Systems, 9(3), 1998, pp. 319–332.

[11] S.M. Bohte, H. La Poutre, and J.N. Kok, "Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks," IEEE Transactions on Neural Networks, 13(2), 2002, pp. 426–435.

[12] X. Tao and H.E Michel, "Data clustering via spiking neural networks through spike timing dependent plasticity," IC-AI, 2004, pp. 168–173.

[13] D.T. Pham, M.S. Packianather, E.Y.A. Charles, "A novel self-organised learning model with temporal coding for spiking neural networks," Intelligent Production Machines and Systems 2006, D.T. Pham, E.E. Eldukhri, A.J. Soroka, editors, Elsevier, Amsterdam, Netherlands, 2006, pp. 307-312.

[14] Pham, D. T. , Packianather, M. S. and Charles, E. Y. A.: Control chart pattern clustering using a new self-organizing spiking neural network; Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, Volume 222, Number 10, 2008; pp 1201-1211.

[15] W. Maass, "Lower bounds for the computational power of networks of spiking neurons," Neural Computation, 8(1), 1996, 1-40.

[16] Spiking Neuron Models. Single Neurons, Populations, Plasticity. Wulfram Gerstner and Werner M. Kistler Cambridge University Press, August 2002.

[17] A. Jahnke, U. Roth and T. Schönauer, "Digital simulation of Spiking Neural Networks," in Pulsed Neural Networks, W. Maass and C.M. Bishop, Eds, The MIT Press, Cambridge, 2001, pp. 237-257.

[18] Bache, K. & Lichman, M., UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science, 2013.

[19] S. Haykin, Neural Networks - A comprehensive foundation (2nd edn). Prentice hall, New Jersey, 1999.