



City Research Online

City St George's, University of London

Citation: Reljan-Delaney, M. & Wall, J. (2018). Solving the linearly inseparable XOR problem with spiking neural networks. In: 2017 Computing Conference. (pp. 701-705). IEEE. ISBN 9781509054435 doi: 10.1109/sai.2017.8252173

This is the accepted version of the paper.

This version of the publication may differ from the final published version. To cite this item please consult the publisher's version.

Permanent repository link: <https://openaccess.city.ac.uk/id/eprint/33579/>

Link to published version: <https://doi.org/10.1109/sai.2017.8252173>

Copyright and Reuse: Copyright and Moral Rights remain with the author(s) and/or copyright holders. Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge, unless otherwise indicated, provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way. For full details of reuse please refer to [City Research Online policy](#).

Solving the Linearly Inseparable XOR Problem with Spiking Neural Networks

Mirela Reljan-Delaney and Julie Wall

School of Architecture Computing and Engineering
University of East London
London, United Kingdom
u1307319@uel.ac.uk, J.Wall@uel.ac.uk

Julie Wall

School of Architecture Computing and Engineering
University of East London
London, United Kingdom
J.Wall@uel.ac.uk

Abstract—Spiking Neural Networks (SNN) are third generation neural networks and are considered to be the most biologically plausible so far. As a relative newcomer to the field of artificial learning, SNNs are still exploring their own capabilities, as well as dealing with the singular challenges that arise from attempting to be computationally applicable and biologically accurate. This paper explores the possibility of a different approach to solving linearly inseparable problems by using networks of spiking neurons. To this end two experiments were conducted. The first experiment was an attempt in creating a spiking neural network that would mimic the functionality of logic gates. The second experiment relied on the addition of receptive fields in order to filter the input. This paper demonstrates that a network of spiking neurons utilizing receptive fields or routing can successfully solve the XOR linearly inseparable problem.

Keywords—Spiking Neural Networks; Receptive Fields; XOR; Boolean Logic; Leaky Fire and Integrate Model

I. INTRODUCTION

This paper presents a comparison of two methods utilising spiking neural networks (SNN) for classification of the linearly inseparable logical exclusive OR (XOR) problem. We explore the possibilities of classification using either only networks of spiking neurons or with the addition of receptive fields (RF), and compare the results for effectiveness and plausibility. The objective of the exercise was to successfully classify the XOR set by creating a SNN in Matlab through implementation of the leaky integrate and fire (LIF) spiking neuron model.

This paper is organised as follows. Section 2 discusses the main components of the networks, i.e. the LIF neurons, RFs and Boolean logic. The first experiment of a SNN model which can produce the functionality of an AND gate is outlined in Section 3 while Section 4 presents the second experiment which successfully classifies the XOR input set using SNNs together with RFs. Lastly, in Section 5 we present our conclusions.

II. SPIKING NEURAL NETWORKS (SNNs)

SNNs are third generation neural networks and as such are a relative newcomer to the field of machine learning. In the past it has been presumed that neurons communicate by passing information encoded in firing rates, called frequency coding, where only the rate of spike intervals is used for the encoding of the information [7, 16]. However, more recent advances in the study of neural communication have revealed that the precise timing of spikes plays a deciding role [4]. Pougam-Moisy and Bohte [15] described the biological basis of SNNs, where each spike represents an action potential, the electric impulse that travels to the neuron and acts upon it by raising its membrane potential. In the biological neuron, the membrane potential is the difference in charge between the negative, potassium charged inside, and the sodium positively charged outside [20]. One of the most significant ways SNNs differ from the second generation artificial neural networks is that the network does not fire with each individual input. Instead, pre-synaptic neurons fire series of action potentials which raise the membrane potential of the receiving neuron, also called the postsynaptic neuron. If the membrane potential reaches a set threshold, the neuron fires by emitting a series of spikes itself [5]. Figure 1 depicts a simplified biological neuron with a series of input spikes, the membrane potential of the neuron and the corresponding output spike.

Even though all neurons seem to function on the same principle, variations in the behaviour of the neurons have provided the basis for different models [7]. One of the more popular neuron models is the LIF, which is the basis of our SNN and discussed next.

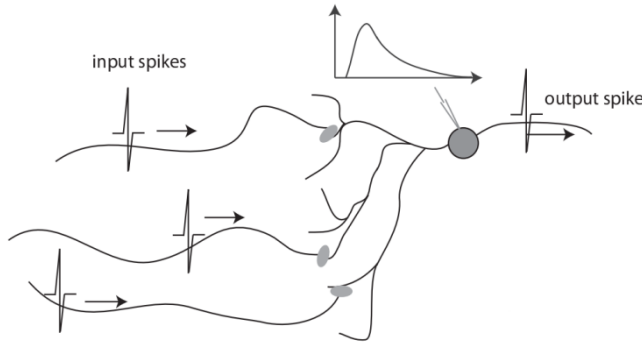


Fig. 1. Model of a biological neuron. Input spikes are traveling down dendrites towards the soma. Inside the soma they combine and upon reaching the threshold the soma discharges an action potential, an output, which travels down the axon [7]

A. Leaky Fire and Integrate (LIF) Neuron Model

According to Pougam-Moisy and Bohte [15], the LIF model is a neuron model that has been derived from the Hodgkin-Huxley model, a complex SNN model that aims to re-create the function of the giant nerve fibre in the squid [9, 10]. However, the LIF model has an advantage over the Hodgkin-Huxley model in that it is simpler and easier to manipulate.

The role of the leakage in the LIF model is to insure that the neuron does not retain accumulated voltage in cases when it has not reached the threshold and fired, as doing so would produce false results when a later group of inputs arrives. The LIF model combines computational simplicity with efficiency which makes it popular with the research community and our model of choice [14]. The LIF model follows the formula:

$$\tau \frac{d(v)}{d(t)} = -v(t) + RI(t) \quad (1)$$

where τ represents the membrane time constant, v is the membrane potential, R is the membrane resistance with the input $I(t)$. Once v reaches a threshold, the membrane potential is reset to a basic initial value called the reset potential. This is followed by a refractory period which is a time period immediately following the release of the action potential during which the neuron will be unresponsive to any stimulus.

B. Receptive Fields (RF)

According to Alonso and Chen [2] in the context of neural stimulation, RFs have first been described by Heartline [8] in work discussing the response of nerve fibres to light stimulation. He observed that when the light falls within a certain region of the retina, stimulation occurs. As Glackin [6] explains, we now know that RFs consist of two fields of influence, an ‘inner’ region facilitates stimulation while the ‘outer’ region inhibits the inputs that it receives, see Figure 2.

The purpose of this process is to isolate the distinctive properties of the input set. Each input has a RF attached which enables designated frequency detection. This is significant as certain frequencies will be representative of a particular attribute and the receiving neuron will become a detector of said attribute. Bohte, La Poutre and Kok [3] comment that the possible number of distinct inputs in a SNN with RFs is only limited by the fields

filtering ability. This can become clearer if we look at the RF formula:

$$k_{ij} = e^{-((x_m - y_o)/d_m)^2} \quad (2)$$

where k_{ij} is a scalar variable which will modify the output spike train frequency of the related neuron, x_m is the operating frequency of the RF, y_o is the input spike train frequency to the RF and d_m denotes the width of the RF. By narrowing the width of the RF it is possible to increase the number of RFs in the network which enables the network to isolate a greater number of specific frequencies. RFs can be successfully employed as a part of the solution for linearly inseparable sets, as demonstrated by Stromatias [20] and later in this paper where we describe a network which will rely on RFs to successfully classify the linearly inseparable set XOR.

C. Boolean Logic

The basis of all computer operations is Boolean logic and is based on only two values: zero (off) and one (on) [12]. Logic gates control current flow and can be organised in configurations which help to solve given problems. These configurations can be very complex, but they all consist of some basic building blocks. This is a powerful concept and our intention is to see whether the principles of Boolean logic can be combined with SNNs.

In first generation artificial neural networks, the threshold is a simple binary step function, an artificial neuron in its simplest form. Rosenblatt [17] improved on this design by creating a single-layer perceptron which has non-fixed weights and is capable of learning. Minsky and Papert [13] highlighted the shortcomings in this solution by demonstrating its limitations. In particular, it was shown that a single layer perceptron is not capable of solving linearly inseparable problems, such as the XOR problem. A linearly inseparable outcome is the set of results, which when plotted on a 2D graph cannot be delineated by a single line. A classic example of a linearly inseparable problem is the XOR function and this has resulted in XOR becoming a benchmark problem for testing neural network capabilities in solving complex problems.

There have been some attempts to integrate binary logic and logic gates with SNNs. Vogels [22] explored the accidental creation of logic gates in unsupervised network models which use synfire chain propagation, a synchronized excitation of multiple neurons in a feed-forward network with many layers [1]. This work is interesting from the standpoint of proving the possibility that Boolean logic is acting itself out in neural systems and would merit further investigation by the process of reversed engineering. On the other hand, Tam [20] discusses the differences between binary encoding and encoding information in time-dependent spiking frequencies. Tam’s mathematical work proposes substitution of multi-input AND-gates and multi-input OR-gates with a new concept of the MIN-gate, which is based on the majority/minority rule voting system. It is a promising concept, but the significance of the paper lies more in the correct identification of some of the challenges which lie in translating binary code to spiking code and furthering the understanding of how these challenges can be overcome. While Tam [21] successfully explores the problem theoretically this paper outlines the challenges encountered during the practical implementation of the concept.

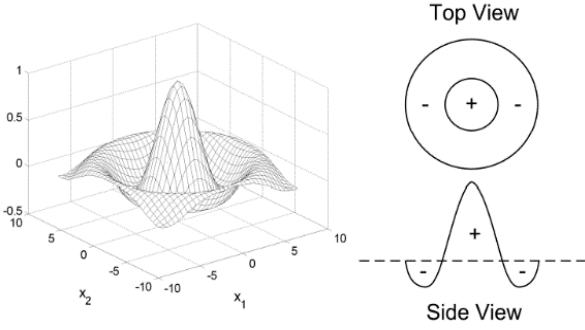


Fig. 2. RF model [6] Left: A 3D model of the RF where the middle conical region is the excitatory field and the surrounding depression the inhibitory area. The input which falls within the excitatory region is promoted, while the input that falls outside it is depressed. Right Top view: The same RF viewed from the above. The circle with the plus sign is the area covering the desired frequency, while the surrounding area with the minus sign corresponds to frequencies outside the specified range. Right Side view: Cross-section of the RF model

III. SNN MODEL OF THE AND FUNCTION

As XOR is considered one of the most challenging Boolean logic functions, we chose the AND function as the starting point for testing the possibility of creating a network of spiking neurons to perform logic gate operations in order to, ultimately, correctly classify the XOR data set. Fig. 3 is the visual representation of the design of this SNN with two inputs, A and B. The inputs will carry frequencies representative of the Boolean values Zero and One. Even though it would be possible to model an AND neuron which will fire only if two inputs arrive simultaneously, we have opted to explore a specific representation of these values using assigned frequencies. The values we have chosen are 51Hz to represent the Boolean One and 25Hz to represent the Boolean Zero. These values are chosen as they are decisively different from each other and can be clearly represented visually. Other frequencies can be used in the future. The input frequencies are encoded into linear spike trains, i.e. the value of the distance between the action potentials, known as the inter spike interval (ISI), is a constant. The network was designed to take advantage of the precise timing between action potentials. If the ISIs throughout spike train input A is in synch with the ISIs in input B, this will signify that both inputs have the same frequency.

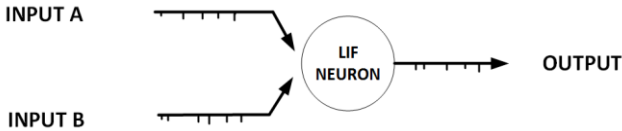


Fig. 3. Network diagram for the AND SNN which consists of two spike train inputs, A and B, each of which can be encoded to represent a Boolean value

In our experiments, we use linear inputs as they are computationally easier to process, and give more predictable results even though they are not realistic representations of neural inputs [6]. The parameters of the network were set to ensure that the neuron should produce an output which corresponds to the assigned frequencies for the Boolean values. Once the network was implemented, we tested the proposed solution on the four different scenarios as outlined in Table 1.

TABLE I. AND NETWORK RESULTS

AND inputs	Encoded inputs (Hz)	AND output	SNN output	Comments
(1,1)	(51Hz, 51Hz)	1	51Hz	Successful classification
(1,0)	(51Hz, 25Hz)	0	25Hz	Successful classification
(0,1)	(25Hz, 51Hz)	0	51Hz	Network failed to classify
(0,0)	(25Hz, 25Hz)	0	25Hz	Successful classification

The network has successfully classified all inputs except (0, 1). The reason for this is that the network is capable of successful identifying whether the inputs are identical, while it is not capable of distinguishing which input is the highest. As the network defaults without screening to input B if A and B are not synchronized, the result is a classification error in the set (0, 1). This means that the network that identifies AND based on the comparison of the two inputs has not given us the desired results as it can successfully identify only one feature of the set, and that is whether the inputs are identical. For successful classification of any of the Boolean logic gates it is necessary to identify at least two features. It would be possible to introduce a static variable for comparison, but this kind of solution is not biologically justified. However, this experiment has not been in vain as it has helped us to identify the requirements for the successful representation of the XOR logic gate using SNNs. Furthermore, in its simplified form this network can be used to identify identical spike trains, i.e. a coincidence detector. Coincidence detection is present in biological processes such as the detection of a sounds onset [19]. König, Engel and Singer [11] propose that in biological neurons, coincidence detection possibly has even higher significance than signal integration as it is directly related to the signals temporal properties, is more accurate and is faster at producing results.

IV. SNN MODEL OF THE XOR FUNCTION WITH RFs

Solving even the simplest Boolean logic gates can present a serious challenge. What we have identified with the previous experiment is that we need to find a way to filter the inputs in a way that will assist the neuron in identifying a specific feature of the set we are classifying. To do this, we identified the RF as a recognized, biologically plausible way to do just this.

Fig. 4 shows the topology of our fully connected, feed forward SNN consisting of two inputs of linear spike trains. For consistency reasons, as well as clarity, we have decided to continue using the frequency values of 51Hz and 25Hz representing the Boolean values One and Zero respectively. Furthermore, we set the width of the RFs to ensure they did not overlap. As can be seen from the topology diagram, the RFs are positioned on the connections between the input and the hidden layer neurons.

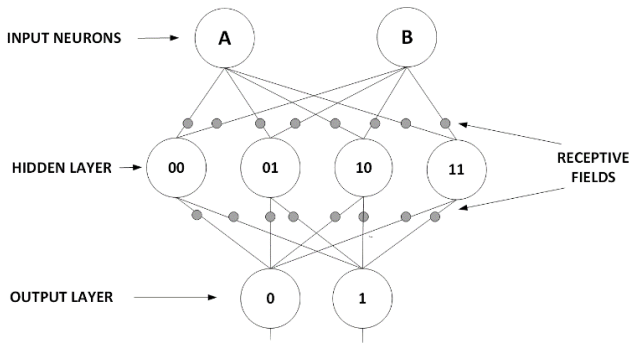


Fig. 4. SNN topology for classification of the XOR data-set using RFs. Input neurons A and B can be encoded in any combination of 0 and 1. The numbering of the hidden layer neurons represents the input combinations that each neuron is set to recognize. The winner takes all method is used for classification in both the hidden and the output layer

All neurons are again encoded using the LIF model. The hidden layer neurons classify the data and the winning neuron (the neuron with the strongest output) is the one that signifies the class of the input at this stage. The output layer receives this signal after it is processed through its own layer of RFs and produces the output spike trains for each output neuron representing the Binary output of Zero or One. The output does not need to correspond to the input frequencies of 51Hz and 25Hz, the neuron with highest number of spikes is the winner and provides classification.

In Fig. 4 we can see that each hidden layer neuron can receive two possible values from the input neurons A and B. These inputs will be either 51Hz or 25Hz, but the RFs on the hidden layer neurons will filter the inputs that are outside their defined frequencies. For example, the RF on input A for neuron 01 has an operating frequency of 25Hz. Similarly, the RF on input B for neuron 01 has an operating frequency of 51Hz, producing an optimized scaled output for the correct set of input frequencies. Any other combination of spike train frequencies passing through these RFs will be diminished before being routed to the hidden layer neurons for processing.

In the interest of comparison, we repeated the same test scenarios we ran on the AND network. Table 2 summarises the inputs, outputs and the successfulness of the XOR classification network. All combinations of XOR inputs were successfully classified by the SNN. Fig. 5 demonstrates the output spike trains produced by each hidden layer neuron when all were passed the input (1, 1) which is equivalent to 50Hz spike trains from both input neurons. The figure demonstrates the correct hidden layer neuron produces the highest firing frequency.

Furthermore, Fig. 4 shows that the output layer neurons receive values from all the hidden layer neurons. These inputs have no predetermined value so the RFs on the output layer neurons are set in the following manner: RFs for the neuron 1 will inhibit inputs outside values that hidden layer neurons 01 and 10 produce, while output neuron 0 will do the same for the hidden layer neurons 00 and 11. For example, the RF on input 01 for neuron 1 has an operating frequency equal to the neuron's 01 output frequency. In this network, as learning is not needed for implementing XOR, the thresholds and weights have to be fine-tuned manually. This is possible as the network is small but in a larger network, learning would need to be implemented to bring about the appropriate set of weights. We have used the work from Glackin [6] as a starting point for further exploration. However, the difference between Glackin's topology and the topology we used is in the connections between the hidden and the output layer. While Glackin implements supervised learning at this point, our experiment inserts RFs.

TABLE II. XOR NETWORK RESULTS

XOR Inputs	Encoded inputs (Hz)	XOR output	SNN Output	Comments
(1,1)	(51Hz, 51Hz)	0	0	Successful classification
(1,0)	(51Hz, 25Hz)	1	1	Successful classification
(0,1)	(25Hz, 51Hz)	1	1	Successful classification
(0,0)	(25Hz, 25Hz)	0	0	Successful classification

As it has been demonstrated, this SNN network has successfully solved the problem of linear inseparability and correctly classified the XOR set. This has shown that RFs are effective at filtering inputs, a very useful feature in SNN processing and can be used as a standalone solution. However, the solution is not perfect. As it stands it is effective for one set of frequencies and to change that would require manual adjustment of the weights. For a successful result, generalizable across a range of input frequencies, we plan to combine both RFs and learning as our next step, in particular spike-timing-dependent-plasticity (STDP). STDP is a learning algorithm where the connection between firing neurons is strengthened or weakened depending on the correlation of the firing times. It has been shown that STDP is able to cope with both linear and more complex Poisson inputs [23]. The use of STDP would allow for an increase in set-size, as well as a variation in input types.

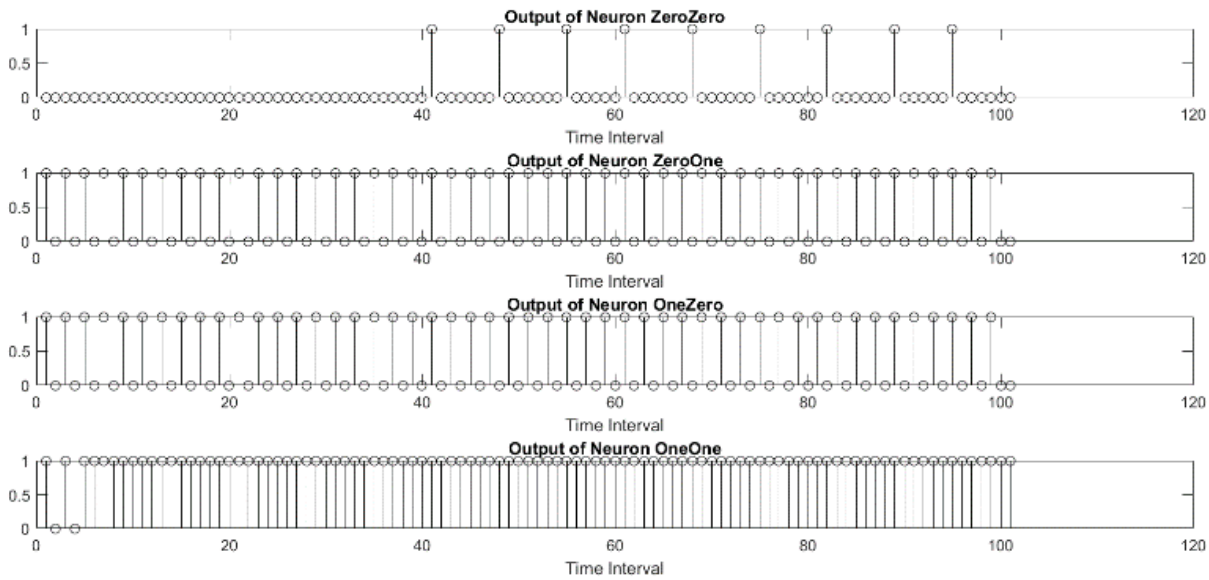


Fig. 5. Outputs of all four hidden layer neurons after the input of 51Hz from neuron A and neuron B. This equates to Boolean input (1, 1). The hidden layer has correctly identified the input the neuron 11 as the winner with the highest output firing frequency

V. CONCLUSION

The first SNN implementation of the AND logic function did not produce the results we expected, but it did sow the seeds for further exploration of non-linear schemes which was the ultimate aim of this work. It also helped crystallize our ideas, and bring into focus the requirements for the XOR network. On the positive side, the AND SNN did work as a coincidence detector, which play a significant role in auditory bilateral synchronization detection.

The second SNN implementation of the XOR logic function did achieve its goal and was successful at classifying this linearly inseparable problem. We have shown that RFs are a powerful aid in the implementation of SNNs and it would be interesting to test their range and effectiveness further, as well as find their limitations. The solution we have presented is scalable, and it can be applied to other datasets, such as the Fisher Iris data set. However, the network is not flexible as the weights have to be fine-tuned. It is our plan to combine RFs with learning as that would create a much more pliable network, capable of adapting itself to solving a wide range of frequencies and problems.

REFERENCES

- [1] M. Abeles, "Synfire chain," Scholarpedia, vol. 4(7), pp. 1441, 2009.
- [2] J-M. Alonso and Y. Chen, "Receptive field," Scholarpedia, vol. 4(1), pp. 5393, 2009.
- [3] 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, vol. 13(2), pp. 426–35, March 2002.
- [4] S.M. Bohte, "The evidence for neural information processing with precise spike-times: A survey," Natural Computing, vol. 3(2), pp. 195–206, 2004.
- [5] B.W. Connors and M.A. Long, "Electrical synapses in the mammalian brain," Annual Review of Neuroscience 27, pp. 393–418, Feb 2004.
- [6] C. Glackin, L. Maguire, L. McDaid and H. Sayers, "Receptive field optimization and supervision of a fuzzy spiking neural network," Neural Networks, vol. 24(3), 247–56, April 2011.
- [7] A. Gruning and S.M. Bohte, "Spiking Neural Networks: Principles and Challenges," in European Symposium on Artificial Neural Networks (ESANN), Computational Intelligence and Machine Learning. Bruges, Belgium, pp. 1-10, 2014.
- [8] H.K. Hartline, "The response of single optic nerve fibres of the vertebrate eye to illumination of the retina," American Journal of Physiology, vol. 121, pp. 400-415, 1938.
- [9] A.L. Hodgkin and A.F. Huxley, "Currents carried by sodium and potassium ions through the membrane of the giant axon of Loligo," The Journal of Physiology, vol. 116(4), pp. 449–472, April 1952a.
- [10] A.L. Hodgkin and A.F. Huxley, "The components of membrane conductance in the giant axon of Loligo" The Journal of Physiology, vol. 116(4), pp. 473–496, April 1952b.
- [11] P. König, A.K. Engel and W. Singer, "Integrator or coincidence detector? The role of the cortical neuron revisited," Trends in Neurosciences, vol. 19(4), pp. 130–137, April 1996.
- [12] C.D.D. Miller, V.E. Heeren and J.E. Hornsby, "Mathematical ideas," 12th edn. Pearson Addison-Wesley, Boston, January 2011.
- [13] M. Minsky and S. Papert, "Perceptions" M.I.T. Press, Oxford, England, 1969.
- [14] R.B Stein, "The frequency of nerve action potentials generated by applied currents" Proceedings of the Royal Society of London. Series B, Biological Sciences, vol. 167(1006), pp. 64-86, January 1967.
- [15] H. Paugam-Moisy and S. Bohte, "Computing with spiking neuron networks," in Handbook of natural computing, pp. 335-376. Springer, Berlin, Heidelberg, 2012.
- [16] F. Ponulak and A. Kasinski, "ReSuMe learning method for Spiking Neural Networks dedicated to neuroprostheses control," in Proceedings of EPFL LATSIS Symposium 2006, Dynamical Principles for Neuroscience and Intelligent Biomimetic Devices, pp. 119-120, 2006.
- [17] F. Rosenblatt, "The Perceptron – a perceiving and recognising automaton," Report 85-460-1. Cornell Aeronautical Laboratory, 1957.s
- [18] L. Robert, C. Naeger, and W. Maass. "What can a neuron learn with spike-timing-dependent plasticity?," Neural computation, vol. 17(11), pp. 2337-2382, 2005.
- [19] L.S. Smith and D.S. Fraser, "Robust sound onset detection using leaky integrate-and-fire Neurons with depressing synapses," IEEE Transactions on Neural Networks, vol. 15(5), pp. 1125–1134, 2004.
- [20] E. Stromatias, "Developing a supervised training algorithm for limited precision feed-forward spiking neural networks," arXiv preprint arXiv:1109.2788., 2011.

- [22] N.D. Tam, "Digital equivalence of biological neural AND-gate, OR-gate and MIN-gate," *International Journal of Computer and Information Technology*, vol. 2(5), 2013.
- [23] T.P. Vogels, "Signal propagation and logic Gating in Networks of Integrate-and-Fire Neurons," *Journal of Neuroscience*, vol. 25(46), pp. 10786–10795, 2005.
- [24] O. Zohar, T.M. Shackleton, I. Nelken, A.R. Palmer and M. Shamir, "First spike Latency code for Interaural phase difference discrimination in the guinea pig inferior Colliculus," *Journal of Neuroscience*, vol. 31(25), pp. 9192–9204, 2011.