# Analysis of Double Threshold Neurons for Spiking Neural Networks

**Clemens JS Schaefer** [1]

## Abstract

Recent advances in Machine Learning, especially regarding DNNs, claim to use the working of the human brain to achieve their astonishing results. However, in a deeper analysis, it turns out that DNNs use only a fraction of the mechanisms the human brain uses to solve problems. SNNs match the working of the human brain closer since the incorporate the time domain. This paper will introduce double threshold neurons for SNNs, which match another mechanism in the human brain - the existence of excitatory and inhibitory signals. Double threshold neurons are able to process both signals. This paper will present an in-depth analysis as well as experimental results. [1]

## 1. Introduction

In the past years, Deep Neural Networks (DNN) became the foundation of many modern advances in artificial intelligence and improved state-of-the-art results in a manifold spectrum from speech recognition over visual object recognition to drug discovery and genomics (LeCun et al., 2015). DNNs fall into the category of brain-inspired algorithms since they supposedly mimic the way how a human brain works. However, realistically modern DNNs only use a very small fraction of techniques humans brain use to work.

One approach to further match the functioning of the brain are spiking neural networks (SNN). The idea of SNNs is derived from the Hodgkin-Huxley model (Hodgkin & Huxley, 1952), which was conceptualized into a framework called the Spike Response Model (SRM) by Gerstner (2001). In contrast to traditional DNNs, SNNs introduce a time component, which means that the computational neurons receive various amounts of signals over time. Moreover, practically a neuron in an SNN integrates over a certain amount of time and forwards a signal via its synapses to connected neurons once a certain threshold is reached. This is an analogy to changes in the potential of a neuron's membrane in the human brain due to the inputs of its dendritic tree. At first, the inputs produce electrical transmembrane currents, which create postsynaptic potentials. Significant postsynaptic potentials can lead to the creation of an action potential or spike by utilizing the voltage-sensitive channels embedded in the neuronal membrane. The spikes are then propagated to other neurons via the axon (Izhikevich, 2007).

Spiking neural nets are interesting in various aspects. Firstly they come with more components to infer information, e.g. the spike timing and are more biologically realistic (Tavanaei et al., 2018). Furthermore, Merolla et al. (2014) showed that Synaptic Operations (SynOp) are potentially way more energy efficient than multiply-accumulate operations used by many state-of-the-art neural nets, this suggests that spiking neural nets might be very suitable for hardware implementations.

Nevertheless spiking neural networks also only partially capture the workings of human neurons. So far spiking neural networks consider excitatory inputs, which bring a neuron closer to a (positive) threshold. However, not all important signals in the brain are excitatory. There are also inhibitory inputs, which reduce membrane potential and hence in case of a positive threshold the likelihood of generating a spike (Kandel et al., 2000).

Furthermore, there are bipolar neurons in the human cerebral system. Bipolar neurons are used for vision (in the retina), olfactory sensing and hearing. Their unique property is that they have two distinct specialized processes, the dendrite to receive electrical signals and the axon to transmit signals. Additionally, the bipolar neurons of the retina have the receptive-field around them and enabling them to be excitatory or inhibitory (Kandel et al., 2000).

This work will introduce and analysis double threshold neurons for computational neural networks. Double threshold neurons will be able to process and forward excitatory and inhibitory inputs, which can be seen as an analogy to bipolar neurons in humans. And hence represent a novel approach to closer match the brains functioning for computational neural networks.

[1] Department of Computer Science and Engineering, University of Notre Dame, Notre Dame, IN 46556. Correspondence to: Clemens JS Schaefer <cschaef6@nd.edu>.

[1] The source code for this project can be found under: https://github.com/clee1994/DTN

## 2. Related Work

Underlying to all SNN are different neuron models. In the early nineteen-fifties, Hodgkin and Huxley developed a neuron model after observing the electric currents in giant neurons of squids. Their model consists out of four coupled nonlinear differential equations and hence is rather difficult to analyze mathematically (Hodgkin & Huxley, 1952). Various researchers, later on, have tried to simplify this model and came up with different systems of two equations (Abbott & Kepler, 1990). Another adaptation by Gerstner (2001) reduces the Hodgkin Huxley system to just a single variable $u(t)$ representing the membrane potential of a neuron, this model is called the Spike Response Model (SRM). The SRM can be simplified to the integrate and fire model without spatial structure. This work will analyze the double threshold neurons in the framework of a integrate and fire model (IF). The following description of the integrate and fire model follow closely Gerstner (2001).

The circuit of an integrate and fire model is built of a capacitor $C$ parallel to a resistor $R$ driven by a current $I(t)$, this current can be divided into two components:

$$I(t) = I_R + I_C \tag{1}$$

The resistive current $I_R$ can be computed by Ohm's law as:

$$I_R = u/R \tag{2}$$

with $u$ as the voltage applied to the resistor. The current which charges the capacitator $C$ can be derived using the definition of capacity $I_C = q/u$, with $q$ as the charge and $u$ the voltage.

$$I_C = C\frac{\partial u}{\partial t} \tag{3}$$

Hence 2 and 3 can be plugged into 1:

$$I(t) = \frac{u(t)}{R} + C\frac{d}{dt}u \tag{4}$$

Further, we can introduce the membrane time constant $\tau = RC$ which can be introduced into 4 by multiplying with $R$ and then substituting $RC$ with $\tau$.

$$\tau\frac{d}{dt}u = -u(t) + RI(t \tag{5}$$

Equation 5 is the standard form the integrate and fire model, where $u(t)$ stands for the membrane potential. The IF model does not explicitly state the form of an action potential, moreover, the spikes are treated as formal events completely characterized by a firing time $t^{(f)}$, which is a threshold process.

$$u(t) = \vartheta \Longrightarrow t = t^{(f)} \tag{6}$$

After the threshold is reached and the neuron "fires" the membrane potential is reset to its reset value $u_r$ (mostly assumed to be zero for simplicity).

$$\lim_{\delta \to 0_+} u(t^{(f)} + \delta) = u_r \tag{7}$$

For $t > t^{(f)}$ the membrane potential follows equation 5 again, until the threshold is reached again.

Using the I&F model it is possible to derive the interspike interval or firing rate of a neuron given a certain input. Gerstner (2001) illustrated that with the example of a constant current input $I(t) = I_0$. Therefore we have to integrate 5 with the initial condition $u(t^{(0)}) = u_r$ assuming that the first spike has occurred at $t = t^{(0)}$. Equation 5 is a linear first order differential equation and can be solved using an integrating factor. The next steps will sketch the process of doing so:

$$\frac{du}{dt} + \frac{1}{\tau}u(t) = \frac{1}{\tau}RI(t) \tag{8}$$

$$\mu(t)\frac{du}{dt} + \mu(t)\frac{1}{\tau}u(t) = \mu(t)\frac{1}{\tau}RI(t) \tag{9}$$

After bringing the equation into the right form and introducing the integrating factor, it is possible to analyze the integrating factor in a bit more detail assuming that 10 holds true.

$$\mu(t)\frac{1}{\tau} = \mu'(t) \tag{10}$$

$$\frac{1}{\tau} = \frac{\mu'(t)}{\mu(t)} \tag{11}$$

$$\frac{1}{\tau} = ln(\mu(t))' \tag{12}$$

$$\int \frac{1}{\tau}dt = ln\mu(t) + m \tag{13}$$

$$\mu(t) = me^{\int \frac{1}{\tau}dt} \tag{14}$$

We can substitute 10 in 9, which yields:

$$\mu(t)\frac{du}{dt} + \mu'(t)u(t) = \mu(t)\frac{1}{\tau}RI(t) \tag{15}$$

$$(\mu(t)u(t))' = \mu(t)\frac{1}{\tau}RI(t) \tag{16}$$

$$\mu(t)u(t) + l = \int \mu(t)\frac{1}{\tau}RI(t)dt \tag{17}$$

$$u(t) = \frac{\int \mu(t)\frac{1}{\tau}RI(t) - l}{\mu(t)} \tag{18}$$

Hence we already found an expression for $\mu(t)$ (equation 10) we can state 21 as follows:

$$u(t) = \frac{\int me^{\int \frac{1}{\tau}dt}\frac{1}{\tau}RI(t) - l}{me^{\int \frac{1}{\tau}dt}} \tag{19}$$

$$u(t) = \frac{\int e^{\int \frac{1}{\tau}dt}\frac{1}{\tau}RI(t) - m}{e^{\int \frac{1}{\tau}dt}} \tag{20}$$

Using our membrane time constant $\tau = RC$ and the fact that we only integrate from 0 till the time we reach the threshold at time $t - \hat{t}$ further simplifies the equation.

$$u(t) = \frac{\int e^{\frac{t-\hat{t}}{\tau}dt}\frac{1}{C}I(t) - m}{e^{\frac{t-\hat{t}}{\tau}}dt} \tag{21}$$

Now using the initial condition $u(t^{(0)}) = u_r$ results in:

$$u(t) = u_r \exp - \frac{t-\hat{t}}{\tau} + \frac{1}{C}\int_0^{t-\hat{t}} \exp - \frac{s}{\tau} I(t-s)ds \tag{22}$$

In case we assume a static current input $I(t) = I_0$ and $u_r = 0$ equation 22 becomes:

$$u(t) = RI_0(1 - \exp{(-\frac{t-t^{(0)}}{\tau})}) \tag{23}$$

For $t \xrightarrow{\text{inf}}$ the membrane potential approaches $RI_0$, in case $RI_0 < \vartheta$ the neuron will never "fire". However for $RI_0 < \vartheta$ the neuron will reach its threshold regularly.

$$\vartheta = RI_0(1 - \exp{(-\frac{t-t^{(0)}}{\tau})}) \tag{24}$$

The firing rate or the inter spike interval can be computed by setting 23 equal to $\vartheta$ (as in 24) and solving for $t - t^{(0)}$ yields:

$$t - t^{(0)} = \tau ln(\frac{RI_0}{RI_0 - \vartheta}) \tag{25}$$

Given a set of neurons and their connections the weights or strength of the connections must be learned to provide a useful neural network, e.g. to detect handwritten digits. Traditional neural networks use the backpropagation algorithm to learn weights, which needs to be adapted for spiking neural networks. Many researchers studied learning rules for spiking neural networks without hidden layers, for example, Pfister et al. (2006) used a supervised learning paradigm to develop a synaptic update rule, which utilizes gradient ascent to maximizes the likelihood of postsynaptic firing at one or several desired firing times.

Ponulak & Kasiński (2010) propose a learning rule called ReSuMe (or Remote Supervised Method) based on the Widrow-Hoff rule by minimizing the error between the target and output signals. They found a way to do that without an explicit need to compute the gradient. Similarly, Mohemmed et al. (2012) presented SPAN another learning rule established on the Widrow-Hoff rule. Their idea is to transform spike trains during the learning phase into analogue signals and perform all mathematic operations on the converted signals. Furthermore Gilra & Gerstner (2017) came up with a learning rule for Feedback-based Online Local Learning Of Weights (FOLLOW), where the error is fed back through fixed random connections with a negative gain in order for the network to learn desired dynamics. Further Maass et al. (2002) used SNNs to solve classification tasks by utilizing the p-delta rule to train the readout layers of a liquid state machine.

Nonetheless, the papers mentioned so far were solely concerned about SNNs with no hidden layers, but recent advances have shown that Deep Neural Networks are indeed very powerful and can achieve astonishing results and hence SNNs with multiple layers are probably necessary to achieve state-of-the-art accuracies. Until now only a few scientists have investigated SNN with hidden layers, mostly due to difficulties of finding the derivatives necessary to perform a proper backpropagation. Bohte et al. (2002) presented a supervised training rule for SNNs with hidden layers, by reformulating the problem in a way that all equation are differentiable and hence standard gradient descent can be performed. One shortcoming of this algorithm is that it can't start learning from a quiescent state. Sporea & Grüning (2013) introduced an extension of the ReSuMe for multiple hidden layers which can overcome the shortcoming of Bohte's algorithm. This extension back propagates error signals linearly. One of the most recent learning rules for multilayer SNNs came from Huh & Sejnowski (2017). Instead of hard spiking non-linearities, they used "soft" spiking thresholds for which standard techniques of gradient descent are applicable and hence enabling standard backpropagation.

In the same year as Huh & Sejnowski (2017), Zenke & Ganguli (2017) presented SuperSpike which is another learning rule for the training of multi-layer spiking neural networks with deterministic integrate-and-fire neurons.

Apart from learning rules researchers analyzed various details of SNNs, for instance, Neftci et al. (2016) investigated the effects of stochasticity in SNNs. Inspired by biological studies of the human brain Neftci et al. (2016) introduced multiplicative noise on the weights and hereby randomly omitting the transmission of signals. They were able to achieve on par results with other SNNs and discovered that their learned networks were remarkable sparse and robust to pruning as well as quantization.

SNNs have improved drastically over the last years, Beyeler et al. (2013) were able to achieve an error rate of 8% on the MNIST dataset after 100 rounds of random sub-sampling using a hierarchical spiking neural network that integrates a low-level memory encoding mechanism with a higher-level decision process. In 2015 Querlioz et al. accomplished a 6.3% error on the same dataset utilizing models of non-volatile memory devices as synapses and STDP learning. In the same year Diehl & Cook (2015) reached an error rate of 5% by the means of conductance-based synapses, spike-timing-dependent plasticity with time-dependent weight change, lateral inhibition, and an adaptive spiking threshold. Comparing those results to other types of neural networks, for example, DropConnect (Wan et al., 2013) with an error rate of 0.21% suggest that SNNs can still be further improved and a remain a relevant field of research.

## 3. Analysis

This work will present a design, an in-depth analysis as well as an implementation of double threshold neurons. And hence will propose a novel approach of utilizing the biological concept of inhibitory and excitatory spikes for machine learning.

As shown in the Related Work section one of the most fundamental neuron models is the integrate and fire model. Thus the double threshold neurons will be introduced in the framework of the I&F model. Equation 5 remains valid and the main change of the double threshold neurons consists of adding another threshold $\vartheta_2$ to the model. In case the membrane potential equals or exceeds the additional threshold $\vartheta_2$ it will "fire" an inhibitory signal to its connected neurons. Hence the following equations represent I&F model with double threshold neurons:

$$\tau \frac{d}{dt} u = -u(t) + RI(t \tag{26}$$

$$\begin{aligned} u(t) = \vartheta_1 \Longrightarrow t = t^{(f)} \\ u(t) = \vartheta_2 \Longrightarrow t = t^{(f)} \end{aligned} \tag{27}$$

$$\lim_{\delta \to 0_+} u(t^{(f)} + \delta) = u_r \tag{28}$$

For simplicity, we will assume from now on that $\vartheta_1 = -\vartheta_2$ (in future we might lose this assumption to allow for a broad model and more complicated neural networks). Since equation 5 doesn't change, equation 22 remains true to for the double threshold neurons. Analyzing the interspike interval will reveal further details of the inner function of the double threshold neurons. Giving a constant current input the neuron will behave in the same way as ordinary single threshold neurons with one exception. We stated before that for a constant current input nothing will happen in case $RI_0 < \vartheta_1$, this is not true for the double threshold neurons. Here we have two thresholds hence the neuron will only remain quiescent as long as $RI_0 > \vartheta_2$ and $RI_0 < \vartheta_1$ are full filled concurrently. For all other cases the neuron will fire according to equation 25, where $\vartheta$ is either $\vartheta_1$ or $\vartheta_1$ depending on the value $RI_0$.

Things become more interesting when $I(t)$ is not constant any more and rather is a random process as we would expect it under the real circumstance. Usual the inter-spike interval is assumed to be a Poisson process. Nonetheless, a pure Poisson process to govern $I(t)$ would be insufficient to fully discover the behaviour of double threshold neurons. Hence a Bernoulli process to govern the direction of the spike, as in whether it is inhibitory or excitatory, is introduced. In order to find out the inter-spike interval for a double threshold neuron given a Poisson Bernoulli input, we need to solve equation 22. Let us assume $u_r = 0$, which eliminates the first part of the equation. Since $I(t)$ is not static anymore, rather a random variable, the integral cannot easily be solved anymore. It is necessary to employ integration by parts. The integral of a Poisson Bernoulli random variable is difficult to evaluate, hence the author assumed that with a big enough variance $\lambda$ in the Poisson process and a small enough spike size, the Poisson Bernoulli random variable can be approximated with a Gaussian random variable. Whereby the mean $\mu$ of the Gaussian can be seen as the mean $p$ Bernoulli part and the variance $\sigma^2$ of the Gaussian as the variance $\lambda$ of the Poisson part. The integral from zero to a fixed value is given as follows:

$$\int_0^{t-\hat{t}} I(t)dt = \frac{1}{2}\pi erf(t-\hat{t}) \tag{29}$$

This allows us to apply integration by parts to 22 and assuming $u_r =$ hence yields:

$$\begin{aligned} u(t) = \frac{1}{C} \exp{(-\frac{t-\hat{t}}{\tau})} \frac{1}{2}\pi erf(t-\hat{t}) - \\ \frac{1}{C}\int_0^{t-\hat{t}} \exp{(-s/\tau)} \frac{1}{2}\pi erf(t-\hat{t}) ds \end{aligned} \tag{30}$$

This expression can be further simplified to:

$$u(t) = \frac{1}{2C}\pi erf(t-\hat{t})(\exp(-\frac{t-\hat{t}}{\tau}) - \exp(-\frac{t-\hat{t}}{\tau})+1) \quad (31)$$

$$u(t) = \frac{1}{2C}\pi erf(t-\hat{t}) \quad (32)$$

Given equation 32 we are able to to determine $t-\hat{t}$ again.

$$t-\hat{t} = erf^{-1}(\frac{2Cu(t)}{\pi}) \quad (33)$$

Realistically however it is unlikely that $I(t)$ will behave like a standard Gaussian random variable, that's why it is necessary to incorporate the mean and variance.

$$t-\hat{t} = erf^{-1}(\frac{2Cu(t)}{\pi\sigma} - \mu) \quad (34)$$

By substituting $u(t)$ with either $\vartheta_1$ or $\vartheta_1$ in 34 it is possible to compute the inter-spike interval for excitatory and respectively inhibitory spikes.

## 4. Evaluation

Given the analytical solution in the Analysis section, we want to verify in this section that the proposed double threshold neurons indeed behave as analyzed. Therefore we used the Python package Brian2 to simulate the neurons.

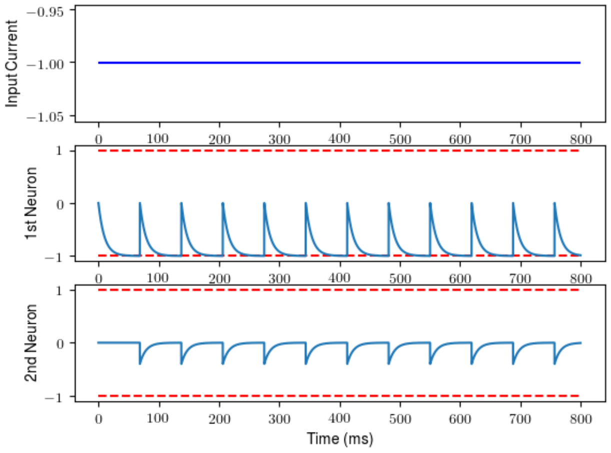

*Figure 1.* Static input current

Figure 1 shows a static negative current input in the upper graph and the membrane potential of a double threshold neuron in the middle. The bottom section illustrates the membrane potential of another neuron which is connected to the neuron in the middle section. We can see clearly that the static current is lower than $\vartheta_2$ of the neuron and hence we can observe a constant inter-spike interval.

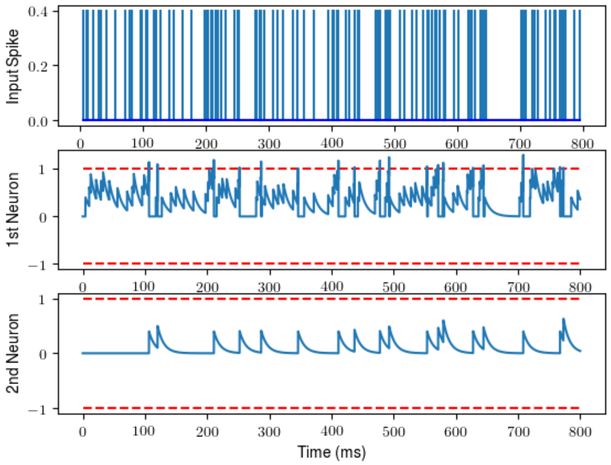

*Figure 2.* Poisson input current

The next figure (2) has the same structure and content as Figure 1 with the difference that the input are now spikes with inter-spike intervals distributed according to a Poisson distribution. Notably is that the membrane potentials are not more erratic and the inter-spike interval of the first neurons potentially mimics a Poisson process as well.

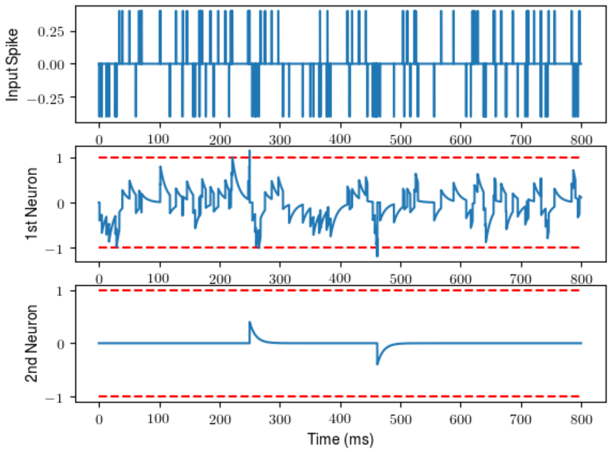

*Figure 3.* Poisson Bernoulli input current

Figure 3 displays the membrane potential and input spikes of double threshold neurons which are actually exposed to a Poisson-Bernoulli input. Here we can clearly see how the input is governed by a Bernoulli random variable which decided whether to emit an excitatory spike or an inhibitory one. We can see how the membrane potential can evolve in both directions and that the neuron is able to “fire” excitatory and inhibitory spikes.

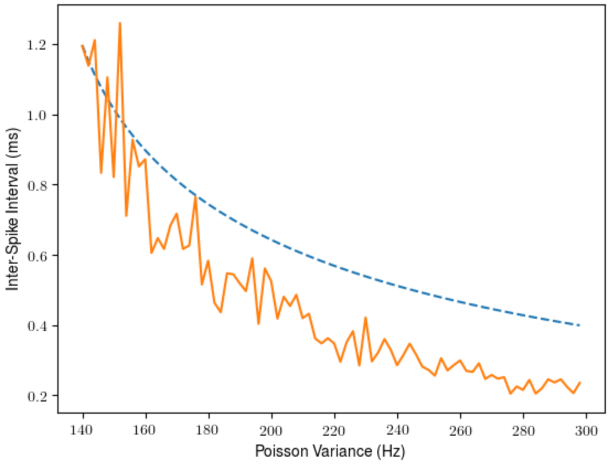

*Figure 4.* Inter-Spike Interval with varying Poisson Variance $\lambda$ for Excitatory Spikes (orange continuous line experimental results and blue dashed line analytic results)

Figures 1, 2 and 3 illustrated clearly what happened with one single neuron over the course of time but they give little information whether our analytically obtain inter-spike interval formula is actually correct. Therefore we simulated the behaviour of a neuron over a much longer interval of time and computed the inter-spike interval given by our experimental data and using our analytic solution.

Figure 4 shows the inter-spike interval for excitatory spikes analytically and experimentally for a varying Variance $\lambda$ of the Poisson process. Figure 5 displays nearly the same as Figure 4 with the marginal difference that that here the negative spikes are subject of the analysis. It becomes clear that with a larger variance the inter-spike interval declines, in both cases. This result makes sense especially considering that the mean $\mu$ is zero and hence the more variance the more often a threshold is actually meet or exceeded.

Figures 6 and 7 also represent the inter-spike interval for excitatory and inhibitory spikes respectively. Here the Poisson variance was kept the same and the Bernoulli mean $\mu$ was varied. The two graphs seem to exhibit an interesting behaviour. For the excitatory spikes a bigger $\mu$ means a smaller inter-spike interval, meanwhile, for inhibitory spikes, the reverse is true. This behaviour also makes sense since the $\mu$ can be seen as a kind of drift term which will push toward a boundary.

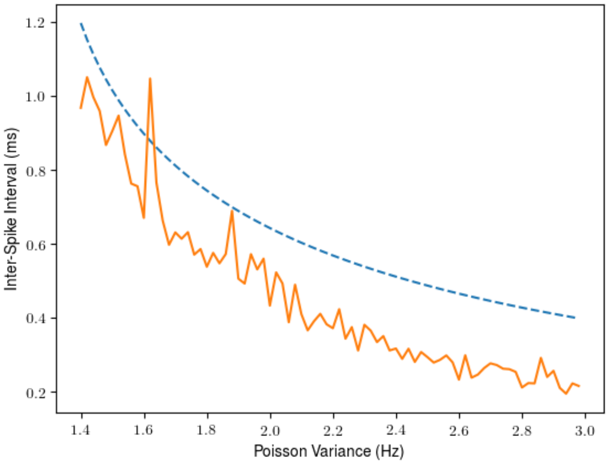

*Figure 5.* Inter-Spike Interval with varying Poisson Variance $\lambda$ for Inhibitory Spikes (orange continuous line experimental results and blue dashed line analytic results)

Additionally, to evaluate whether the proposed bipolar neurons are useful and capable of learning, they could be employed to perform classic machine learning tasks, such as recognizing handwritten letters. However, developing an actual learning rule and implementing it is beyond the scope of this paper. Nonetheless, it is insightful to observe what happens when double threshold neurons receive real input from MNIST (handwritten digit recognition). The data set for this task was initially put together by LeCun et al. (1998) and served as a benchmark for many machine learning algorithms such as DNNs. The MNIST dataset consists of 60,000 training images and 10,000 testing images each 28 by 28 pixels. Although there exists justified criticism that the MNIST dataset is too easy to meaningfully evaluate modern Machine Learning algorithm, for SNNs it still is a challenging task as shown by the results achieve of prior work (see Related Work section: around 5% error rate of SNNs in contrast to 0.21% for DNNs).

Since the functioning of SNNs is intrinsically different to other supervised learning algorithms the MNIST dataset needs to be customized. SNNs require timed spikes which are not provided by static images. Those recordings are usually rare to find and hence data sets need to recorded instead of just collected and labelled.

In 2015 Serrano-Gotarredona & Linares-Barranco presented two data sets for event-driven object recognition using a Dynamic Vision Sensor (DVS). The first was the Poker-DVS data set which was created by browsing through a specially made poker card decks in front of a DVS camera for 24s, whereby each card appeared for 20-30ms on the screen. The second is the MNIST-DVS which contains 30,000 DVS camera recordings which were generated by displaying 10,000

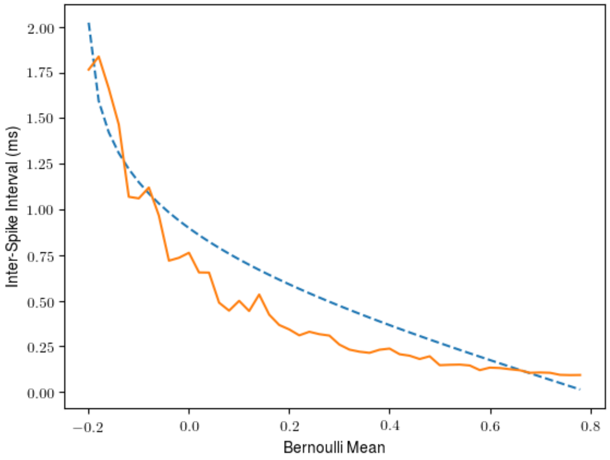

*Figure 6.* Inter-Spike Interval with varying Bernoulli Mean $p$ for Excitatory Spikes (orange continuous line experimental results and blue dashed line analytic results)

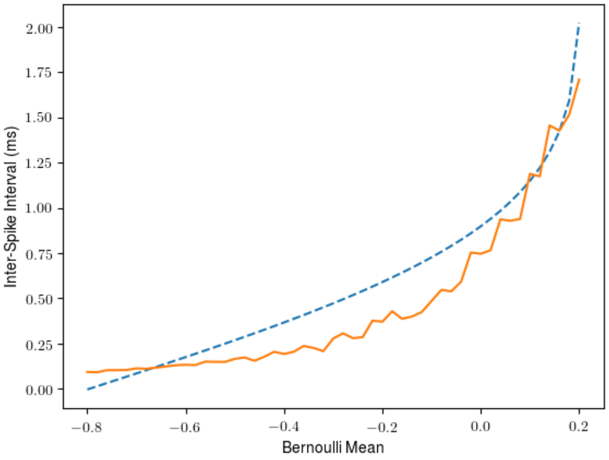

*Figure 7.* Inter-Spike Interval with varying Bernoulli Mean $p$ for Inhibitory Spikes (orange continuous line experimental results and blue dashed line analytic results)

moving symbols from the standard MNIST 70,000-picture database on an LCD monitor for about 23s each (every image was displayed at three different scales, hence 30,000 recordings).

Orchard et al. (2015) created the N-MNIST by using an actuated pan-tilt camera platform and converted the MNIST and Caltech101 to a neuromorphic vision dataset. They argue that moving the sensor instead of the image in the screen is a more biologically realistic simulation of sensing and eliminates timing artefacts introduced by monitor updates.

Following Orchard et al.s argument the author decided to use the N-MNIST dataset. Figure 8 illustrates a six from the N-MNIST data set, whereby the signal was added over a time span. Note that red pixels represent excitatory signals and blue pixels inhibitory signals, meanwhile, black pixels indicate no activity during the observed time span. Additionally, it should be observed that there is noise involved in the images and the pixels barely form a complete six.

The image was flattened and feed into a simple SNN with double threshold neurons. Since as of now there is no learning rule, all weights had an equal value of one. The network had an input layer with 1156 neurons (width multiplied by the height of the image), a single hidden layer with 50 neurons and the output layer had 10 neurons. All layers were fully connected to the subsequent layer. Figure 9 shows an excerpt of the activity of one neuron from the last layer. It is visible that both inhibitory and excitatory spikes are received which shows that double threshold neurons might indeed be beneficial for SNNs.

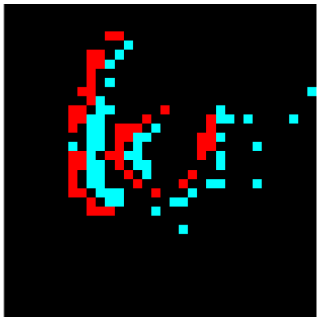

*Figure 8.* Sample illustration for a six from the N-MNIST

## 5. Conclusion

This paper introduced double threshold neurons in the context of the integrate and fire model. The double threshold neurons are inspired by bipolar neurons in the human brain which are able to receive and transmit excitatory and inhibitory signals. An analysis of the double threshold neurons was performed, which showed that given some assumptions and simplification it is possible to compute the firing rate of a double threshold neuron which is exposed to a Poisson-Bernoulli input. Further, this paper implemented the double threshold neurons in Python utilizing the Brian2 package. Experiments illustrated the membrane potential of the introduced neurons given various different inputs. Finally, the experimental behaviour was compared against the analytical solution and it was displayed that the analytic solution indeed matches the experimental results. A higher variance in the Poisson process leads to the smaller inter-spike interval and the mean can influence whether there are more

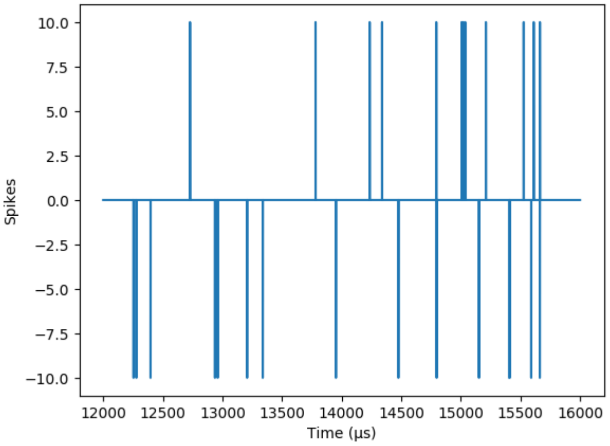

*Figure 9.* Sample illustration of a neuron from the last layer of simple SNN, which received input from the N-MNIST

inhibitory or excitatory spikes.

The work of this paper is far from complete and needs some thorough checking of its mathematics. The author intends to keep working on the subject at hand which includes making the math more sound and potentially develop a learning rule so that double threshold can be finally used for SNN, which can be trained to solve real-world problems.

## References

Abbott, LF and Kepler, Thomas B. Model neurons: From hodgkin-huxley to hopfield. In *Statistical mechanics of neural networks*, pp. 5–18. Springer, 1990.

Beyeler, Michael, Dutt, Nikil D, and Krichmar, Jeffrey L. Categorization and decision-making in a neurobiologically plausible spiking network using a stdp-like learning rule. *Neural Networks*, 48:109–124, 2013.

Bohte, Sander M, Kok, Joost N, and La Poutre, Han. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002.

Diehl, Peter U and Cook, Matthew. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in computational neuroscience*, 9:99, 2015.

Gerstner, Wulfram. A framework for spiking neuron models: The spike response model. In *Handbook of Biological Physics*, volume 4, pp. 469–516. Elsevier, 2001.

Gilra, Aditya and Gerstner, Wulfram. Predicting non-linear dynamics by stable local learning in a recurrent spiking neural network. *Elife*, 6:e28295, 2017.

Hodgkin, Alan L and Huxley, Andrew F. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–544, 1952.

Huh, Dongsung and Sejnowski, Terrence J. Gradient descent for spiking neural networks. *arXiv preprint arXiv:1706.04698*, 2017.

Izhikevich, Eugene M. *Dynamical systems in neuroscience*. MIT press, 2007.

Kandel, Eric R, Schwartz, James H, Jessell, Thomas M, of Biochemistry, Department, Jessell, Molecular Biophysics Thomas, Siegelbaum, Steven, and Hudspeth, AJ. *Principles of neural science*, volume 4. McGraw-hill New York, 2000.

LeCun, Yann, Bottou, Léon, Bengio, Yoshua, and Haffner, Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

LeCun, Yann, Bengio, Yoshua, and Hinton, Geoffrey. Deep learning. *nature*, 521(7553):436, 2015.

Maass, Wolfgang, Natschläger, Thomas, and Markram, Henry. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural computation*, 14(11):2531–2560, 2002.

Merolla, Paul A, Arthur, John V, Alvarez-Icaza, Rodrigo, Cassidy, Andrew S, Sawada, Jun, Akopyan, Filipp, Jackson, Bryan L, Imam, Nabil, Guo, Chen, Nakamura, Yutaka, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

Mohemmed, Ammar, Schliebs, Stefan, Matsuda, Satoshi, and Kasabov, Nikola. Span: Spike pattern association neuron for learning spatio-temporal spike patterns. *International journal of neural systems*, 22(04):1250012, 2012.

Neftci, Emre O, Pedroni, Bruno U, Joshi, Siddharth, Al-Shedivat, Maruan, and Cauwenberghs, Gert. Stochastic synapses enable efficient brain-inspired learning machines. *Frontiers in neuroscience*, 10:241, 2016.

Orchard, Garrick, Jayawant, Ajinkya, Cohen, Gregory K, and Thakor, Nitish. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9:437, 2015.

Pfister, Jean-Pascal, Toyoizumi, Taro, Barber, David, and Gerstner, Wulfram. Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural computation*, 18(6):1318–1348, 2006.

Ponulak, Filip and Kasiński, Andrzej. Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting. *Neural computation*, 22(2):467–510, 2010.

Querlioz, Damien, Bichler, Olivier, Vincent, Adrien Francis, and Gamrat, Christian. Bioinspired programming of memory devices for implementing an inference engine. *Proceedings of the IEEE*, 103(8):1398–1416, 2015.

Serrano-Gotarredona, Teresa and Linares-Barranco, Bernabé. Poker-dvs and mnist-dvs. their history, how they were made, and other details. *Frontiers in neuroscience*, 9:481, 2015.

Sporea, Ioana and Grüning, André. Supervised learning in multilayer spiking neural networks. *Neural computation*, 25(2):473–509, 2013.

Tavanaei, Amirhossein, Ghodrati, Masoud, Kheradpisheh, Saeed Reza, Masquelier, Timothee, and Maida, Anthony S. Deep learning in spiking neural networks. *arXiv preprint arXiv:1804.08150*, 2018.

Wan, Li, Zeiler, Matthew, Zhang, Sixin, Le Cun, Yann, and Fergus, Rob. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning*, pp. 1058–1066, 2013.

Zenke, Friedemann and Ganguli, Surya. Superspike: Supervised learning in multi-layer spiking neural networks. *arXiv preprint arXiv:1705.11146*, 2017.