Final Project Proposal: SNN for Image Classification

Authors: SHENG-QIAN LI, JIA-JUN LAI, NAI-REN SONG, FANG-KAI HSIAO

1. ABSTRACT

This project is dedicated to improving the accuracy of Spiking Neural Network (SNN) models by exploring various learning rules and encoding methods in conjunction with the Diehl and Cook model for image classification. Inspired by the paper titled "Spiking Neural Networks for Image Classification Project by Osaze Shears and Ahmad Hossein Yazdani," we have identified the significant potential of the Diehl and Cook model. As a result, our primary focus in this project is to optimize the Diehl and Cook Network Model.

2. INTRODUCTION

Spiking Neural Networks (SNNs), the third generation of neural networks, offer a closer approximation to natural neural networks compared to traditional Artificial Neural Networks (ANNs).

(As shown in Fig 1: ANN&SNN)

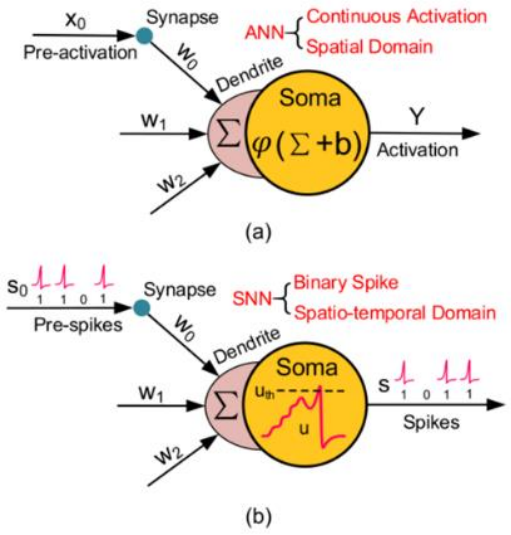


Fig 1: (a) the activity of ANN neuron; (b) the activity of SNN neuron

Unlike ANNs, SNNs transmit data as discrete spikes of 0 or 1. Neurons in SNNs only transmit information when their membrane potential reaches a threshold, mimicking the selective firing observed in biological neurons. The impact of transmitted spikes is delicately shaped in SNNs, subtly influencing other neurons by adjusting their membrane potential through synapse weights. Crucially, SNNs integrate the concept of time during execution, capturing temporal dependencies absent in ANNs. The event-driven property of SNNs emphasizes the significance of triggered events in shaping dynamic neural network behavior. This unique combination of spike-based communication, threshold firing, synapse-weight modulation, temporal awareness, and event-driven processing positions Spiking Neural Networks as a promising paradigm for more biologically plausible and temporally nuanced artificial intelligence.

Table 1: Traditional ANNs vs SNNs

Network	Execution Time	Neuron I/O	Output Mechanism
ANN	Instantaneous	Raw Numerical Value	Activation Function
SNN	Duration of Time	Binary Spike Value	Threshold Value

SNNs boast energy and area efficiency through selective, event-driven information transmission. As shown in Fig 2, the digital design of SNN doesn't need multipliers to do multiplications, which save computations, energy and chip areas.

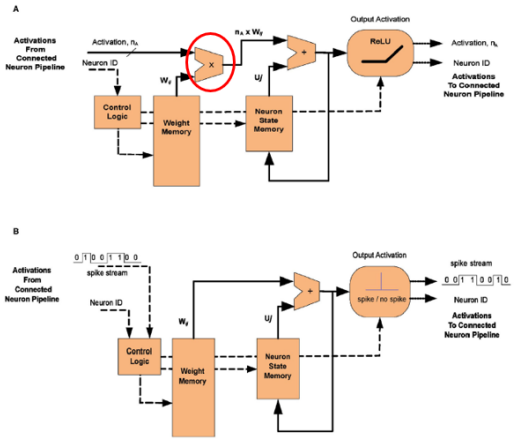


Fig 2: A. Hardware design of ANN neuron; (b) harware design of SNN neuron

The efficient on-chip learning minimizes computational complexity, while fault tolerance is enhanced by the discrete property of spikes. Overall, SNNs stand out for their versatility and advantages in energy-efficient computing and biologically inspired artificial intelligence.

In the field of computer vision, especially in classification tasks, the performance of SNNs currently lags far behind that of the well-established CNNs, our motivation is to enhance accuracy in image classification. We intend to make SNNs adaptable to various datasets and improve them by exploring new encoding methods, innovative models, and advanced learning approaches.

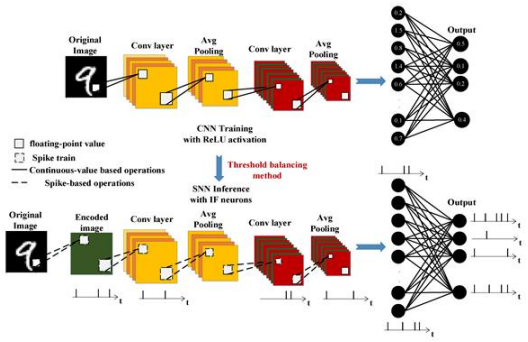


Fig 3: an example of image classification using CNN and SNN

3. REVIEW OF PREVIOUS WORK

In this section, we first review the results of the midterm report, primarily replicating the findings of the paper without altering hyperparameters. Additionally, we conducted experiments with a neuron model not tested in the paper, specifically the AdaptiveLIF model, under three encoding schemes and three learning rules.

The results indicate that our training outcomes are consistent with those presented in the paper. During the training process, we found that RankOrder Encoded Input takes the most time to train, and Bernoulli is the fastest, and Bernoulli's accuracy is higher than RankOrder. This is different from the intuitive feeling of the model. The more complex the model, the longer it usually takes to train. However, more complex models tend to have better performance at the extreme limit of accuracy.

The results reveal that the best outcomes were achieved with Poisson encoding and the DieAndCook model, as well as the Adaptive model. Therefore, in this project, we focus on modifying the hyperparameters of these two models.

Table 2: Classification Accuracy for Poisson Encoded Input

Neuron Model	Learning Rule	Our Accuracy	Paper Accuracy
IF	PostPre(STDP)	0.10	0.10
IF	WeightSTDP	0.10	0.09
IF	Hebbian	0.10	0.10
SRM0	PostPre(STDP)	0.10	0.10
SRM0	WeightSTDP	0.10	010
SRM0	Hebbian	0.10	0.10
DiehlAndCook	PostPre(STDP)	0.81	0.79
DiehlAndCook	WeightSTDP	0.77	0.79
DiehlAndCook	Hebbian	0.82	0.80

Table 3: Classification Accuracy for Bernoulli Encoded Input

Neuron Model	Learning Rule	Our Accuracy	Paper Accuracy
IF	PostPre(STDP)	0.10	0.10
IF	WeightSTDP	0.10	0.10
IF	Hebbian	0.10	0.10
SRM0	PostPre(STDP)	0.10	0.10
SRM0	WeightSTDP	0.10	0.09
SRM0	Hebbian	0.10	0.10
DiehlAndCook	PostPre(STDP)	0.39	0.37
DiehlAndCook	WeightSTDP	0.37	0.34
DiehlAndCook	Hebbian	0.32	0.35

Table 4: Classification Accuracy for RankOrder Encoded Input

Neuron Model	Learning Rule	Our Accuracy	Paper Accuracy
IF	PostPre(STDP)	0.10	0.10
IF	WeightSTDP	0.10	0.10
IF	Hebbian	0.10	0.10
SRM0	PostPre(STDP)	0.10	0.10
SRM0	WeightSTDP	0.10	010
SRM0	Hebbian	0.10	0.10
DiehlAndCook	PostPre(STDP)	0.13	0.14
DiehlAndCook	WeightSTDP	0.13	0.11
DiehlAndCook	Hebbian	0.15	0.13

4. TECHNICAL PART

4-1. Neuron Models

(1) Integrate-and-Fire (IF) Model

IF Model describes the membrane potential of a neuron in terms of the synaptic inputs and the injected current that it receives. An action potential (spike) is generated when the membrane potential reaches a threshold.

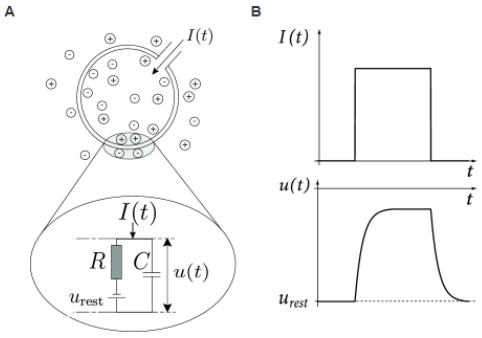


Fig 4: **A.** A neuron, which is enclosed by the cell membrane (big circle), receives a (positive) input current I(t) which increases the electrical charge inside the cell. The cell membrane acts like a capacitor in parallel with a resistor which is in line with a battery of potential u_{rest}. **B.** The cell membrane reacts to a step current (top) with a smooth voltage trace (bottom).

(2) Leaky Integrate-and-Fire (LIF) Model

The LIF neuron model introduces the concept of leaky current based on the IF model, simulating the leakage of the neuronal membrane potential. When the membrane potential reaches the threshold, a spike is still generated. However, after the spike, the membrane potential briefly decreases, mimicking the phenomenon of continuous leakage of the neuronal membrane potential.

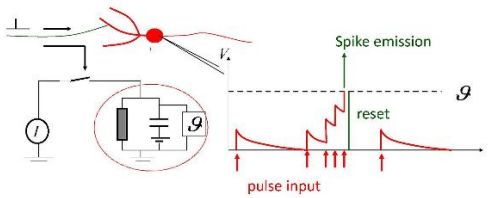


Fig 5: A neuron is represented by an RC circuit with a threshold. Each input pulse causes a short current pulse. Voltage decays exponentially. If the threshold is reached an output spike is generated and the volt

(3) Adaptive LIF Model

The Adaptive LIF model introduces adaptive mechanisms based on the standard LIF model to

capture additional features of real neuronal behavior. The adaptive mechanism is typically an adaptive current or conductance, simulating the changes in the neuron's state after firing a spike, making it less likely to generate consecutive spikes in a short period.

(4) DiehlAndCook Model

The DiehlAndCook model typically refers to a neural network model used to describe spiking neurons. The development of this model aims to better understand information processing and learning mechanisms in biological neural networks, focusing on exploring how neurons in biological neural networks process information through learning and shaping. The characteristics of this model may involve dynamic properties of spiking neurons, synaptic plasticity, and other relevant aspects.

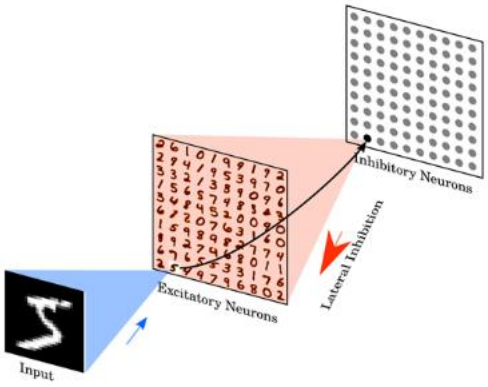


Fig 6: NETWORK ARCHITECTURE. The intensity values of the 28 × 28 pixel MNIST image are converted to Poisson-spike and fed as input to excitatory neurons in an all-to-all fashion. The blue shaded area shows the input connections to one specific excitatory example neuron. Excitatory neurons are connected to inhibitory neurons via one-to-one connections. The red shaded area denotes all connections from one inhibitory neuron to the excitatory neurons.

Different part between DieAndCook and AdapativeLIF is the code below:

```
# Choose only a single neuron to spike.
if self.one_spike:
    if self.s.any():
        _any = self.s.view(self.batch_size, -1).any(1)
        ind = torch.multinomial(
            self.s.float().view(self.batch_size, -1)[_any], 1
        )
        _any = _any.nonzero()
        self.s.zero_()
        self.s.view(self.batch_size, -1)[_any, ind] = 1
```

Diehl And Cook introduced a new flag called self-one spike, which indicates whether only one neuron is allowed to spike in a single time step. After checking whether a neuron spike an action potential, we select only one neuron to spike. Each excitatory layer neuron was connected to only one neuron in the inhibitory layer. On the other hand, the output of the neurons in the inhibitory layer were connected to all the neurons of the excitatory layers except for the one it received input from. Other excitatory neurons were inhibited once an excitatory neuron emitted a spike. An inhibitory neuron inhibited the internal voltage of a neuron once it received a spike. This was helpful since an active neuron could prevent the voltage of other neurons from increasing.

4-2. Update Rule

(1) PostPre

The rule is based on the timing difference between presynaptic and postsynaptic spike occurrences, thereby influencing the connection weights between neurons. If the presynaptic neuron fires a spike before the postsynaptic neuron, it is termed a "pre-post" relationship; conversely, if the postsynaptic neuron's spike occurs after the presynaptic neuron's spike , it is referred to as a "post-pre" relationship.

When the postsynaptic neuron's spike follows the presynaptic neuron's spike, it

typically results in the enhancement of synaptic weights, which reinforcing connections between neurons that exhibit favorable timing relationships.

(2) Hebbian

The learning rule emphasizes the correlation and simultaneous activity between neurons, suggesting that synaptic connections between neurons that fire together should be strengthened. This can be conceptualized as follows, "If an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased."

4-3. Input Encoding

(1) Poisson

In this encoding scheme, information is encoded based on the discharge rate (frequency of spikes) of neurons, where the frequency of spikes reflects the intensity of information, and the stochastic nature of Poisson processes implies random neuronal discharge in this coding paradigm.

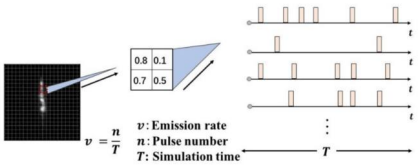


Fig 7: Schematic diagram of Poisson encoding process.

(2) Bernoulli

In Bernoulli rate coding, the discharge rate of neurons is considered, emphasizing the presence or absence of spikes to map information, enabling representation of sparse data with spikes generated only during specific events, in contrast to the frequency-focused Poisson rate coding

(3) RankOrder

Rank Order (Temporal) Coding highlights the temporal order of spikes, utilizing the spike times of various neurons to encode information, emphasizing structural information in the temporal order, which proves crucial in tasks related to representing temporal relationships beyond individual neuron discharge rates.

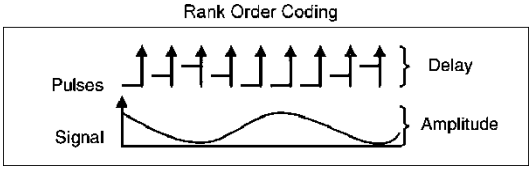


Fig 8 Rank Order Coding. The amplitude of a signal is encoded over time. The higher the amplitude, the shorter the delay (and vice versa).

5. EXPERIMENTS

Due to referencing a paper that focuses solely on optimizing the Poisson encoding method, this project performs internal parameter tuning exclusively for the Poisson encoding approach. The implementation results are presented in tabular form.

Table 4 is pertained to the adjustment of only the batch size, reduced from the original 64 to 32.

Table 5: Classification Accuracy for Poisson Encoded Input

Neuron Model	Learning Rule	Accuracy (modified)	Accuracy (original)
IF	PostPre(STDP)	0.10	0.10
IF	WeightSTDP	0.10	0.10
IF	Hebbian	0.10	0.10
SRM0	PostPre(STDP)	0.10	0.10
SRM0	WeightSTDP	0.10	0.10
SRM0	Hebbian	0.10	0.10
AdaptiveLIF	PostPre(STDP)	0.82	0.81
AdaptiveLIF	WeightSTDP	0.80	0.80
AdaptiveLIF	Hebbian	0.83	0.82
DiehlAndCook	PostPre(STDP)	0.82	0.81
DiehlAndCook	WeightSTDP	0.76	0.77
DiehlAndCook	Hebbian	0.81	0.82

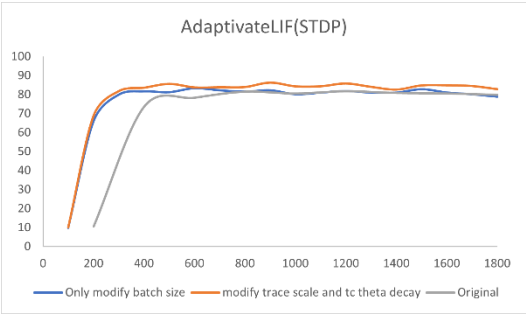
In this project, we were dedicated to optimizing training results by adjusting certain parameters. Table 5 presents the optimized results obtained by not only reducing the batch size but also adjusting the hyperparameters of the Adaptive LIF and DiehlAndCook models in addition.

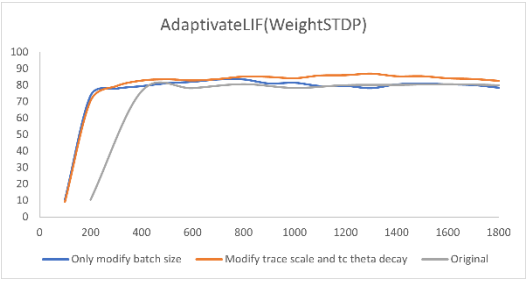
Table 6: Classification Accuracy for Poisson Encoded Input

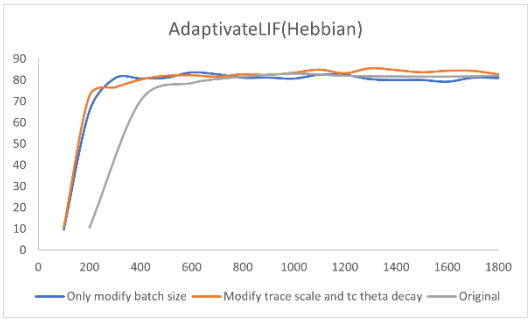
Neuron Model	Learning Rule	Accuracy (modified)	Accuracy (original)
AdaptiveLIF	PostPre(STDP)	0.83	0.81
AdaptiveLIF	WeightSTDP	0.85	0.80
AdaptiveLIF	Hebbian	0.86	0.82
DiehlAndCook	PostPre(STDP)	0.87	0.81
DiehlAndCook	WeightSTDP	0.89	0.77
DiehlAndCook	Hebbian	0.86	0.82

AdaptiveLIF Model:

Followings are the learning curves corresponding to different learning rules for AdaptiveLIF model. Each graph consists of three lines representing before optimization, after optimization (only changing batch size), and after optimization (changing trace scale and tc theta decay).

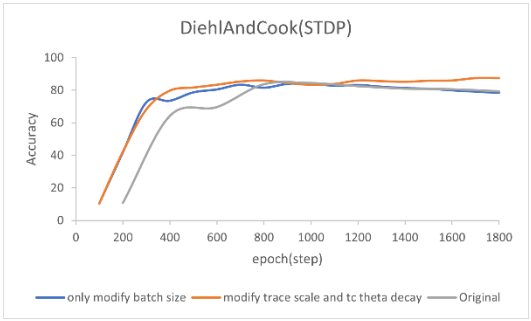


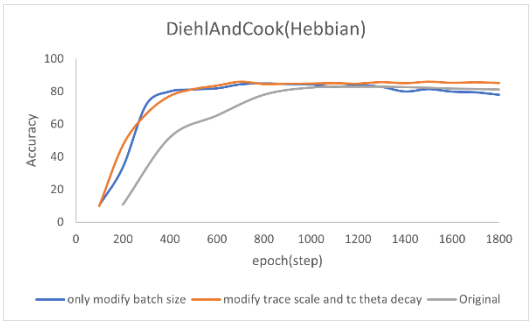




DiehlAndCook Model:

Followings are the learning curves corresponding to different learning rules for DiehlAndCook model. Each graph consists of three lines representing before optimization, after optimization (only changing batch size), and after optimization (changing trace scale and tc theta decay).





The following is an introduction to the hyperparameter we adjusted, along with explanations for the choice of each parameter modification.

```
# ADD BY SELF
trace_scale = 0.5,
tc_theta_decay = 1e5,
```

Trace scale:

Trace scale is a parameter crucial for adjusting the spike trace in neuronal models. Modulating trace scale can have diverse impacts on the learning and behavior of the model:

(1) Sensitivity of Spike Trace

Increasing trace scale enhances the sensitivity of the spike trace, intensifying its reflection of the neuron's spike activity.

(2) Learning Speed of the Model

Given the association of spike trace with synaptic plasticity, elevating trace scale accelerates the model's learning speed, facilitating swift adjustments in synaptic weights and influencing the speed and stability of learning.

(3) Influence on Adaptive Thresholds:

In cases where the model incorporates adaptive thresholds, alterations in the spike trace could affect the speed and stability of adaptive threshold adjustments.

(4) Impact on Hebbian learning rule

Adjusting trace scale can influence the learning rules of Hebbian learning, typically based on the principle of simultaneous activity. Increasing trace scale may strengthen the impact of simultaneous activity between neurons, consequently adjusting the enhancement of synapses.

(5) Impact on STDP learning rule

An increase in trace scale directly influences the strength of spike trace, adjusting how neurons perceive and memorize the timing of pulse activities. A larger trace scale could make neurons more sensitive to the temporal relationships of pulses, consequently influencing STDP calculations and making learning more responsive to simultaneous activity.

Tc theta decay:

In the Diehl and Cook Nodes model, Tc theta decay is utilized to adjust the time constant of the adaptive threshold. This parameter influences the decay rate of the adaptive threshold, thereby impacting the adjustment of neuron thresholds.

(1) Impact on Long-Term Memory in the Model:

A slower decay of the adaptive threshold could result in neurons maintaining memories of past spike activities for an extended period. A faster decay of the adaptive threshold may lead to more transient memory of past activities in neurons.

(2) Influence on Learning Stability:

Adjusting Tc theta decay can affect the learning stability of the model, specifically the impact of the rate of change in neuron thresholds on the learning process.

(3) Impact on Hebbian learning rule

Increasing Tc theta decay may slow down the decay rate of the adaptive threshold, implying a more gradual adjustment of neuron thresholds. Conversely, decreasing Tc theta decay may accelerate the decay of the adaptive threshold, enabling neurons to adapt their thresholds more rapidly.

(4) Impact on STDP learning rule

Adjusting Tc theta decay affects the decay rate of the adaptive threshold in STDP, increasing it may slow down the decay of neurons' adaptive thresholds, thereby influencing STDP calculations.

6. CONCLUSION

Our project thoroughly examines the characteristics and applications of Spiking Neural Networks (SNNs). By adjusting and optimizing the internal parameters with the Poisson encoding method, we show the classification accuracy of various neural models and learning rules under the Poisson encoding method. The optimization results indicate that significant improvements of SNNs' accuracy in tasks such as image classification can be achieved through parameter adjustments, particularly in optimizing the Adaptive LIF and DiehlAndCook models. These findings provide important insights for enhancing SNNs' performance and exploring new encoding methods, innovative models, and advanced learning approaches. Future research can further expand the applications of SNNs and explore additional optimization strategies to facilitate their widespread use in the field of artificial intelligence.

7. REFERNECES

[1] Maass, W. (1997). "Networks of Spiking Neurons: The Third Generation of Neural Network Models." Neural Networks, 10(9), 1659-1671.

[2] Bouvier, M., Valentian, A., Mesquida, T., Rummens, F., Reyboz, M., Vianello, E., & Beigne, E. (2019). "Spiking Neural Networks Hardware Implementations and Challenges: A Survey." ACM Journal on Emerging Technologies in Computing Systems (JETC), 15(2), 1-35.

[3] Li, G., Deng, L., Chua, Y., Li, P., Neftci, E. O., & Li, H. (2020). "Spiking Neural Network Learning, Benchmarking, Programming, and Executing." Frontiers in Neuroscience, 14.

[4] Diehl, P. U., & Cook, M. (2015). "Unsupervised Learning of Digit Recognition Using Spike-Timing-Dependent Plasticity." Frontiers in Computational Neuroscience, 9, 99.

[5] Deng, L., Wu, Y., Hu, X., Liang, L., Ding, Y., Li, G., ... & Xie, Y. (2020). "Rethinking the Performance Comparison Between SNNS and ANNS." Neural Networks, 121, 294-307.

[6] LeCun, Y. (1998). "The MNIST Database of Handwritten Digits."

http://yann.lecun.com/exdb/mnist/.

[7] Hazan, H., Saunders, D. J., Khan, H., Patel, D., Sanghavi, D. T., Siegelmann, H. T., & Kozma, R. (2018). "Bindsnet: A Machine Learning-Oriented Spiking Neural Networks Library in Python." Frontiers in Neuroinformatics, 12, 89.

[8] Osaze Shears, Ahmadhossein Yazdani. (2020). " Spiking Neural Networks for Image Classification ."