Multi-Methods to Model Visual System: Conversion, Training and Encoder Analysis

Mao Chuan
Student ID: 2300013218

Abstract—Spiking neural networks (SNNs) have emerged as a significant research focus in recent years due to their advantages in processing event-driven tasks and performing inference in real-world scenarios. Several approaches have been proposed to achieve high-performance SNNs, such as training deep spiking neural networks using backpropagation or converting pre-trained artificial neural networks (ANNs) into SNNs to enhance accuracy.

Given the unique characteristics of SNNs, performing specific image classification tasks requires transforming pixel-level inputs into spike-based representations over time intervals using encoding algorithms. This study reviews various types of SNNs and examines diverse training and encoding methods in light of these characteristics. To enable a comparative evaluation of different models, a unified neural network architecture is adopted—a simplified visual system network. Additionally, the CIFAR-10 image classification dataset is used to train the network, and the results are thoroughly analyzed.

Index Terms—visual system, spiking neural network, object classification, ANN-to-SNN conversion, training spiking neural networks

I. Introduction

Spiking neural networks (SNNs) are neural networks that exhibit characteristics more closely aligned with biological systems. While the academic community has made remarkable advancements in Artificial Neural Networks (ANNs), SNNs offer distinct and irreplaceable advantages: 1) the ability to process event-driven, time-sequential inputs, 2) energy-efficient transmission of discrete values within the network, and 3) improved interpretability of models.

Due to their low energy consumption, SNNs are well-suited for use as embedded architectures in certain hardware applications. For example, Li Shen et al. proposed an energy-efficient SNN for aerospace object detection tasks by converting Convolutional Neural Networks (CNNs) to SNNs [1]. In ANN-to-SNN conversion, activation layers (e.g. ReLU, Sigmoid) are often replaced with biologically inspired neuron models, such as Leaky Integrate-and-Fire (LIF) neurons. Parameter normalization, which maintains activation levels within biologically plausible ranges and preserves layer-encoded information, plays a crucial role in this process. Weight matrices of linear layers are converted into synapse connection strengths using methods like maximum-value normalization or quantile-based normalization, the latter referred to as RobustNorm.

Bodo Rueckauer et al. demonstrated through mathematical analysis that the spiking rate of biological neurons over time can approximate the activation of ANN layers [2].

$$r_i^l(t) = a_i^l r_{\max} - \frac{V_i^l(t)}{tV_{\text{thr}}} \tag{1}$$

Here, $r_i^l(t)$ is the firing rate of neuron i in layer l, and a_i^l represents the activation of ANN neuron i in the same layer. V_i^l is the membrane potential of layer l, neuron i at time t. When t is sufficiently large, $r_i^l \propto a_i^l$.

Beyond ANN-to-SNN conversion, SNNs can also be trained directly. Unlike ANNs, which utilize continuous values and differentiable activation functions, SNN training faces challenges: 1) non-differentiability due to the Heaviside function in minimizing model loss over parameters; 2) gradient information lose after discretizing layer outputs.

To address non-differentiability, surrogate gradient methods are commonly employed, where surrogate functions approximate gradient descent [3]. Alternatively, Zeroth Order Optimization (ZOO) algorithms, which have a slower convergence speed than First Order Optimization(FOO), compute gradients directly without backpropagation:

$$\nabla f_r(\boldsymbol{x}) \approx \mathbb{E}_{\boldsymbol{z}\sim\lambda}[\frac{n}{r}\left(f(\boldsymbol{x}+r\boldsymbol{z}) - f(\boldsymbol{x})\right)\boldsymbol{z}] \tag{2}$$

where n is the dimensionality of the space, r is random disturbance, $\boldsymbol{z}$ represents samples from distribution λ (e.g. $\mathcal{N}(0,1)$), and $\nabla f(x)$ is used to update parameters (e.g. weights and biases). Bhaskar Mukhoty et al. demonstrated promising results using local zeroth-order optimization to train SNNs directly [4].

Similar to recurrent neural networks, SNNs neurons receive binary pulse stimulation at each time step, transmit signals to subsequent layers, and update their voltage states. This mechanism renders SNNs training time-intensive. Some studies propose novel methods and algorithms to reduce latency and enhance accuracy, such as modifying surrogate functions, adjusting network layers, and employing normalization techniques.

Unlike ANNs for image tasks (e.g. CNNs), which typically accept images directly, SNNs require input over a temporal duration [5]. Some networks repeatedly input pixel-level images, while others utilize complex encoders to transform images into spike sequences over time. Techniques such as Time-to-First-Spike (TTFS) coding and Poisson encoding offer distinct advantages. However, prior studies rarely compare these methods in trained SNNs. Interestingly, a unified layer design: Conv-BatchNorm-IFNode can function as an encoder, allowing pixel-level input.

A successful model effectively captures the behavior of target neurons with a manageable number of parameters. For example, David J. Heeger et al. proposed a computational model for various brain regions [6]. In this study, I designed a biologically inspired visual system to facilitate parameter analysis and SNN comparisons.

The contributions of this research are summarized as follows:

- Propose a Convolutional Neural Network to model the visual system, incorporating the functions of its physiological structures.
- Attempt to convert the CNN-based visual system into a Spiking Neural Network.
- Develop and apply a surrogate gradient method to train SNN with similar architecture.
- Compare the performance of the two networks and evaluate different encoders on the CIFAR-10 dataset.

Previous studies have seldom compared ANNs and SNNs at the same scale, nor have they extensively explored encoder performance. The encoding mechanisms of the visual system are both essential and complex. This work also identifies encoding methods that minimize information loss while aligning with biological characteristics.

Section I introduces the study. A comprehensive review of related works is presented in Section II. Section III outlines the methodology, followed by result analysis in Section IV. Limitations and future work are discussed in Section V, and the study concludes in Section VI.

II. Related Works

Given the capabilities of deep neural networks, most Spiking Neural Networks (SNNs) implementations aim to replicate the behavior of existing Artificial Neural Networks (ANNs) architectures for ease of comparison and to address complex tasks. Tools like SpikingJelly have been widely used for such conversions due to their simplicity and efficiency. SpikingJelly provides useful interfaces, such as the 'Converter' module, which significantly streamlines the conversion process. However, to better understand the underlying conversion methods and access intermediate layer results, I chose to implement the process manually, which facilitates a deeper understanding of the architecture and helps align the structures of SNNs more effectively.

Tong Bu et al. analyzed the limitations of basic ANN-to-SNN conversion frameworks and proposed an optimal conversion method to address these issues [7]. Similarly, Dengyu Wu et al. introduced a new framework that builds on the traditional conversion approach by enabling control over residual currents [8]. Despite these advancements, the basic conversion framework remains a representative paradigm, offering valuable insights into the common conversion processes.

In the context of SNN training, various approaches have been proposed to address the inherent challenges. Beyond the surrogate gradient method, Jun Haeng Lee et al. proposed treating discrete spikes as noise during training, which provides an alternative perspective for improving optimization [9].

While reviewing related works, I also observed a focus on designing surrogate functions tailored to specific tasks. Functions like Sigmoid and Tanh are particularly popular as they better emulate the behavior of ReLU layers in ANNs, enhancing compatibility between the two network types.

These works collectively provide a solid foundation for understanding and improving the ANN-to-SNN conversion process, as well as for designing effective training methodologies for SNNs.

III. Methodology

A. Modeling the Visual System

Several biological studies have elucidated the structure and function of the visual system, as well as the mechanisms of information processing at multiple scales. Due to the complexity of sophisticated neuron models in computational neuroscience—where sometimes a synapse-dendritic model can suffice to accomplish network-level tasks—we opt for a simplified and representative visual model. This approach reduces computational requirements while maintaining the core functionalities of the system.

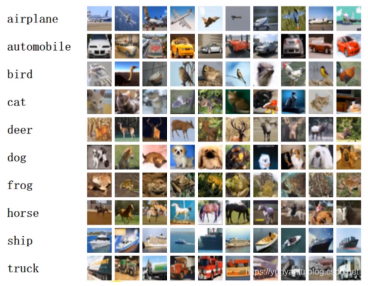
In Fig.1 and Fig.2(a), a Retina-LGN-V1 pathway is proposed to model the visual system. The retina represents the filtering effects of rod and cone cells on varying light intensities and colors. This functionality is implemented as a multi-channel network structure, analogous to the convolutional layers in a Convolutional Neural Network (CNN).

The lateral geniculate nucleus (LGN), a critical intermediate in visual information processing, performs information integration, signal amplification (via the removal of inhibition), and negative feedback regulation. In our network, these functions are captured through two convolutional layers and corresponding activation functions, simulating enhancement and integration processes.

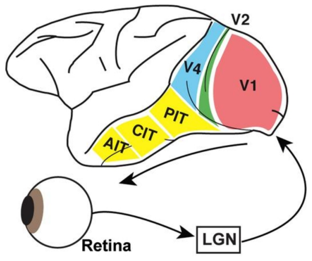
Finally, the V1 region, as a vital part of the visual cortex, is modeled to reflect its advanced information processing capabilities. This is achieved using additional convolutional layers followed by linear layers. The network then flattens the input to produce classification results, completing the visual processing pathway.

B. Conversion to SNN

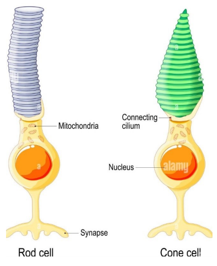
To explore Spiking Neural Network (SNN) conversion, I implemented a conversion method starting from a Convolutional Neural Network (CNN) as shown in Fig. 2(b). The CNN architecture includes convolutional layers (CR,



(a) CIFAR-10



(b) Simple Visual System



(c) Rod and Cone Cells [10]

Fig. 1: Examples from the dataset, showcasing representations of various components including CIFAR-10 images, a simple visual system, and rod and cone cells.

combining Convolution and ReLU) and pooling layers to extract key features.

Initially, I attempted a direct conversion from CNN to SNN, but the resulting accuracy was extremely low, indicating that the SNN failed to adapt to well-trained parameters from the CNN. Later, I found the conversion process requires specific adjustments to certain layers to align with the biological properties of spiking neurons. For instance, the input to the Leaky Integrate-and-Fire (LIF) neuron layer must be processed carefully. In ANNs, Batch Normalization (BatchNorm) layers, which normalize input data to have a mean of 0 and a variance of 1, play a critical role:

$$y(x) = \frac{x - \mathbb{E}(x)}{\sqrt{\mathrm{Var}(x) + \epsilon}} \cdot \gamma + \beta \tag{3}$$

But in the SNNs, inputs are always positive. I absorbed the BatchNorm parameters into the preceding layer to ensure a reasonable output while retaining the trained parameters. Additional tricks were applied, such as normalizing weights and biases using the formulas $\hat{W} = \frac{\lambda_{\text{pre}}}{\lambda} \cdot W$ and $\hat{b} = \frac{b}{\lambda}$ to facilitate smooth parameter transition (where λ_{pre} is maximum input and λ is maximum output). This normalization process enabled the network to maintain its training results during conversion.

Subsequently, I conducted experiments on the converted SNN. The results are presented and analyzed in the next section.

C. Training Similar SNN

Due to time constraints, I opted to use surrogate gradient-based training. During forward propagation, the Heaviside step function was employed, while backpropagation utilized surrogate functions. I experimented with various surrogate functions, including tanh, sigmoid, softsign, and a mixed strategy. A specific surrogate function tested was:

$$g(x) = \frac{1}{2}\left(\frac{x}{1/\alpha + |x|} + 1\right) \tag{4}$$

The performance of these surrogate functions was evaluated based on the prediction accuracy of the same SNN structure, as summarized in Table II.

To further understand the impact of surrogate functions, I compared their similarity to the Heaviside function by plotting their curves. In Fig. 3(a), with $\alpha = 10$, the red curve represents sigmoid, the green curve is softsign, and the purple curve is arctan.

Additionally, I explored fitting common surrogate functions using polynomial approximations via the least squares method. These fitted polynomials could potentially serve as new surrogate functions. For instance, when approximating $f(x) = \frac{1}{1+e^{-20x}}$, the resulting polynomial fit is shown in Fig. 3(b).

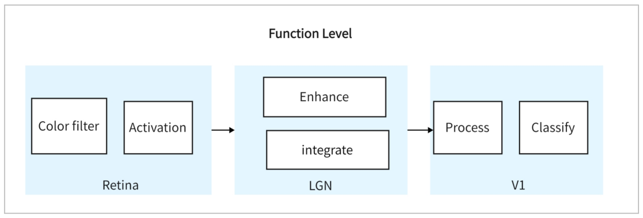
D. Encoder

In my training process, I repeatedly input normalized images *time_steps* times, finding that this approach yields the best results. If the Retina, the first Conv layer, and LIF neurons are considered encoders, this setup constitutes a relatively complex encoder, which is why many works adopt similar architectures.

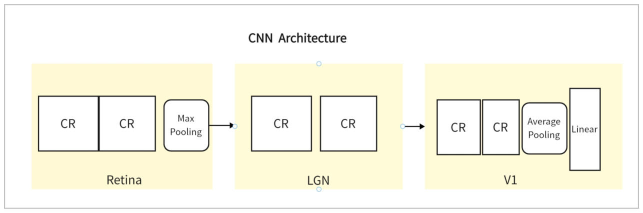
However, more biologically inspired encoding methods, such as Poisson encoding and Time-to-First-Spike (TTFS), may offer better performance in large-scale networks. Below, I present the algorithms for these methods and analyze their characteristics.

Algorithm 1 TTFS Encoding Algorithm

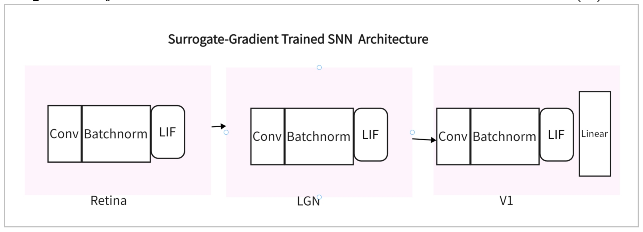
```
Input: Normalized input signal $X = [x_1, x_2, \ldots, x_n]$, where $0 \leq x_i \leq 1$
Output: Spike times $T = [t_1, t_2, \ldots, t_n]$
$T_{\max} \leftarrow$ maximum allowable time
for $i \leftarrow 1$ to $n$ do
    $t_i \leftarrow T_{\max} \cdot (1 - x_i)$
end for
Return $T$
```



(a) Modeling the visual system via primary functions

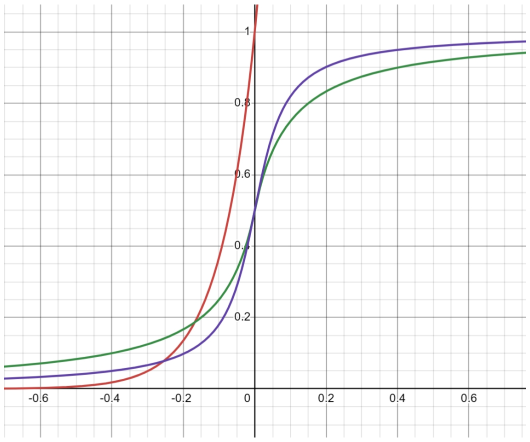


(b) CNN architecture

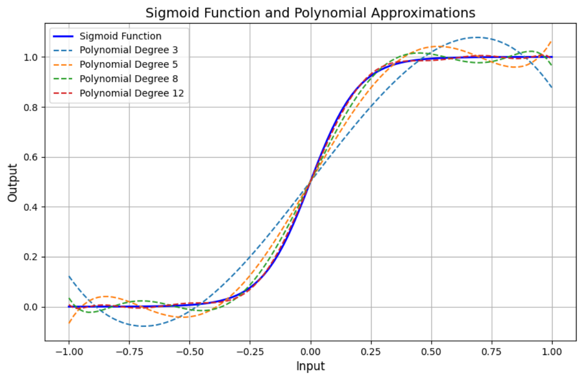


(c) SNN architecture

Fig. 2: Three visual system models. The CNN and SNN architectures emulate specific functions of the biological visual system.



(a) Various surrogate curves



(b) Polynomial fitting for the sigmoid function

Fig. 3: Evaluation of different surrogate functions.

Algorithm 2 Poisson Encoding Algorithm

```
Input: Normalized input signal $X = [x_1, x_2, \ldots, x_n]$, where $0 \le x_i \le 1$
Output: Spike set $S = [s_{i,j}]$, where $s_{i,j}$ indicates a spike (1) or no spike (0) at time step $j$
$f_{\max} \leftarrow$ maximum firing rate
$T_{\text{sim}} \leftarrow$ total simulation time
for $i \leftarrow 1$ to $n$ do
  for $t \leftarrow 1$ to $T_{\text{sim}}$ do
    $p \leftarrow x_i \cdot f_{\max} \cdot \Delta t$ {Spike probability}
    Generate random number $r \in [0, 1]$
    if $r < p$ then
      $s_{i,t} \leftarrow 1$
    else
      $s_{i,t} \leftarrow 0$
    end if
  end for
end for
Return $S$
```

TTFS encoding (Algorithm 1) allows each neuron to spike only once, where stronger stimuli lead to earlier spikes. However, this method is prone to information loss and may not align well with biological retina mechanisms. I propose an improved approach where stronger pixel inputs increase both spike frequency and timing priority, imitating real-world phenomena where higher light intensity draws more attention. This property can be achieved by stacking TTFS encoder and Poisson encoder.

For Poisson encoding (Algorithm 2), the spike probability P^i_{spike} for each pixel x_i is modeled as:

$$P^i_{spike} = P(r \le x_i) \quad (5)$$

Many encoders employ random sampling methods. To evaluate a specific sampling method, we define a pixel as ”well-encoded” if the encoding error between its firing rate and its relative intensity is sufficiently small.

Let $T_{\max}$ be the input time period, $x_{\max}$ the maximum pixel intensity, and num_i the spike count during $[0, T_{\max}]$. Using probability theory:

$$num_i = \sum_{t=1}^{T_{\max}} S_t^i \tag{6}$$

Where:

$$\begin{cases} P(S_t^i = 1) = P_{spike}^i \\ P(S_t^i = 0) = 1 - P_{spike}^i \end{cases} \tag{7}$$

Thus, S_t^i follows a Bernoulli distribution $\mathcal{B}(1, P_{spike}^i)$, and:

$$num_i \sim \mathcal{B}(T_{\max}, P_{spike}^i) \tag{8}$$

We aim to ensure that more pixels satisfy:

$$|num_i - \frac{x_i}{x} T_{max}| \leq \Delta \tag{9}$$

By the Chernoff bound:

$$P(|num_i - P_{spike}^i T_{\max}| \geq T_{\max}\epsilon) \leq 2e^{-2T_{\max}\epsilon^2} \tag{10}$$

This guarantees that pixel x_i is well-encoded when:

$$\frac{x_i}{P_{spike}^i} \geq x_{\max} \tag{11}$$

By adjusting $T_{\max}$, pixels can be adequately represented even after down-sampling, preserving key information while treating irrelevant data as noise. For Poisson encoder, assuming $r \sim \mathcal{U}(0,1)$ and $P_{spike}^i = x_i$, each pixel is encoded proportionally to its intensity, which aligns with biological mechanisms. Designing sampling strategies based on image light intensity ensures effective encoding while discarding extraneous noise.

IV. Result Analysis

A. Results of Multi-Type SNNs

Three types of neural networks were constructed: a Convolutional Neural Network (CNN), a Spiking Neural Network (SNN) converted from the CNN, and a Trained Spiking Neural Network (TSNN). The architecture details are shown in Fig. 2. Table I compares the performance of these networks.

TABLE I: Comparison of testing results for different SNN architectures. The last three rows represent converted SNNs with 1, 2, or all ReLU layers replaced by LIF neurons.

Network Type	Accuracy
Original CNN	64.15
Trained SNN	58.19
Converted SNN (ReLU & BN)	60.10
Converted SNN (1 replaced)	21.19
Converted SNN (2 replaced)	57.96
Converted SNN (all replaced)	52.94

We observe that, compared to the original CNN, both types of SNNs exhibit a decrease in accuracy. This reduction may represent a trade-off for the advantage of improved energy efficiency. Additionally, during the conversion of the CNN to an SNN, ReLU layers were replaced with biologically inspired activation layers incrementally. Interestingly, the accuracy did not decline in a linear fashion, suggesting that certain activation layers play a pivotal role in the network's architecture.

In future applications, selectively converting specific parts of an artificial neural network (ANN) to spiking neural network (SNN) components could be a promising strategy. This approach has the potential to reduce energy consumption while maintaining accuracy.

B. Evaluation of Surrogate Functions

To investigate the impact of different surrogate functions on SNN training, several functions were tested under the same conditions (time steps, learning rate, and number of epochs). The results are summarized in Table II.

TABLE II: Comparison of SNN performance using various surrogate functions.

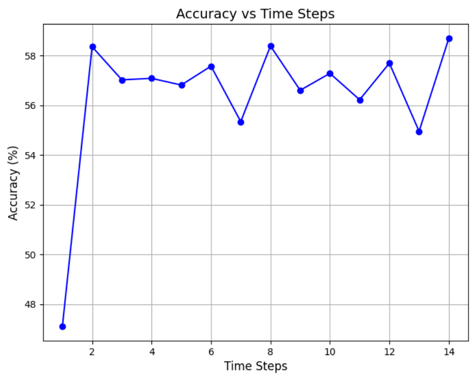
Surrogate Function	Accuracy (%)	Loss
CNN (Baseline)	64.06	0.68
Sigmoid	51.16	1.21
ATan	57.47	0.93
Softsign	58.19	0.89
Mixed Functions	57.86	0.98

The results indicate that SNN training generally underperforms compared to the original CNN. Despite testing various surrogate functions, the trained SNNs showed limited improvements in accuracy, highlighting a fundamental limitation of this approach. Under similar network architectures, the performance of SNNs is generally inferior to that of ANNs. Future work could explore alternative training paradigms or improved surrogate function designs to address this issue.

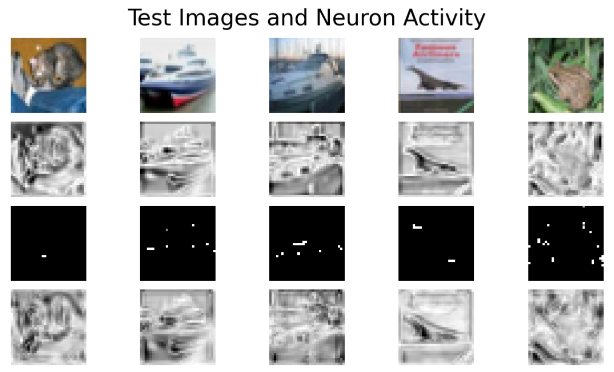
C. Encoder Evaluation

After constructing the SNNs, I tested how time steps effect accuracy (Fig.4 (a)) and printed the spiking patterns at individual moments (Fig.4(b)).

I also evaluated the encoders based on their energy consumption (measured as the average spiking times of the input). Encoders that consume less energy highlight the energy-saving characteristics of spiking neural networks. However, as shown in Table III, reducing energy consumption comes with trade-offs.



(a) The effect of time steps on accuracy.



(b) Intermediate layer output.

Fig. 4: Key findings of spiking neural networks. (a) Accuracy reaches its upper bound after a certain number of time steps. (b) In the multi-channel image representation method, spiking patterns at individual moments are not easily interpretable, but the accumulated information over a time period effectively propagates through subsequent layers.

TABLE III: Evaluation of Different Encoders

Encoder	Energy	Accuracy (%)
Repeat	-	57.47
TTFS	8.10	10.5
Poisson	28.89	21.34
Gaussian	1956	12.6

The results indicate that all randomly sampled encoders exhibited reduced accuracy compared to repeated inputs. This reduction in performance may stem from information loss during the encoding process.

V. Limitations & Future Work

This study represents an initial exploration of building network architectures inspired by the visual system. Since we determined the biologically-based network architecture in the first step, only a limited range of hyperparameters were available for tuning during network training, leading to task performance that fell short of expectations. This limitation is a point of regret.

The primary focus of this research was to investigate the effects of various network architectures and compare different encoding schemes. Despite the constraints, meaningful conclusions can still be drawn based on the relative performance differences observed.

During the implementation of various spiking neural networks (SNNs), an attempt was made to construct a network trained using Zeroth-Order Optimization (ZOO), inspired by the work "A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning" [11]. However, the network failed to learn, likely due to coding errors. Future work will aim to resolve these issues and successfully implement SNNs trained via the ZOO algorithm.

In addition, trainable artificial neural network (ANN)-to-SNN conversion methods, as proposed in [12], present an interesting direction for future exploration. By leveraging these techniques, it may be possible to improve the accuracy of SNNs converted from CNNs.

Another limitation of the current study arises from the biological constraints of the retina, which processes one image stimulus at a time. This restricts the use of larger batch sizes during training. However, given the simplicity of the current network structure, training with a batch size of 1 did not significantly impact results. In future work, a more complex network architecture and a better-organized data structure will be explored to improve training efficiency and performance.

Moreover, for classification tasks, the brain's organizational structure may not rely on a single neural network. It is conceivable that multiple classifiers (e.g. One-vs-One or One-vs-Rest) coexist in certain capacities. Existing studies have revealed that specific types of cells may be responsible for memorizing particular categories of objects [13]. This insight suggests that future attempts to model the brain could benefit from employing more diverse architectures.

VI. Conclusion

This paper is based on a simplified modeling of the visual system and explores various spiking neural networks within a unified framework. It establishes a research paradigm that transitions from constructing biologically abstracted models to implementing them using different network architectures. This approach emphasizes network architecture over performance during the initial model design phase, with performance optimization considered subsequently through various techniques. Such a methodology holds promise for applications in modeling complex biological systems, such as whole-brain modeling, in the future.

This study also provides an in-depth investigation into the performance of spiking neural networks (SNNs) across different architectures and encoding methods. The results reveal that while SNNs offer significant energy-saving potential, their accuracy tends to be lower compared to traditional artificial neural networks (ANNs). This trade-off highlights the challenges of adapting SNNs for practical applications while retaining their advantages in energy efficiency. But through the experiment, I find we can still maintain accuracy by converting some layers.

Key findings include the impact of encoding methods on energy consumption and accuracy, with repeated input encoding yielding the highest accuracy but at the cost of increased energy usage. Given the analysis of the quality of encoders, we can choose some sampling methods to achieve a better balance. Various surrogate functions were also tested, showing that SNN training outcomes are highly sensitive to the choice of surrogate function, yet they remain inferior to the performance of baseline CNNs.

Despite its limitations, this work lays the foundation for future research. Promising directions include improving training methodologies for SNNs, implementing advanced ANN-to-SNN conversion techniques, and addressing the challenges of biological constraints through innovative network designs. By bridging the gap between energy efficiency and accuracy, future studies could unlock the potential of SNNs for real-world applications.

References

[1] Li Shen, Ma Zhong, and Yang Chaojie. ”Conversion of CNN to SNN for Aerospace Object Detection Tasks.” International Conference on Pattern Recognition and Artificial Intelligence, 10.1109/PRAI55851.2022.9904132.

[2] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer and Shih-Chii Liu. ” Conversion of Continuous-Valued Deep Networks to Efficient Event-Driven Networks for Image Classification.” Front. Neurosci. 10.3389/fnins.2017.00682

[3] Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. ”Surrogate Gradient Learning in Spiking Neural Networks: Bringing the power of gradient-based optimization to spiking neural networks.” IEEE Signal Processing Magazine (2019)

[4] Bhaskar Mukhoty, William de Vazelhes et al. ”Direct Training of SNN using Local Zeroth Order Method.” NeurIPS(2023)

[5] Jose Antonio Perez-Carrasco et al. ”Mapping from Frame-Driven to Frame-Free Event-Driven Vision Systems by Low-Rate Rate Coding and Coincidence Processing—Application to Feedforward ConvNets.” IEEE transactions on pattern analysis and machine intelligence (2013)

[6] David J.Heeger, Eero P.Simoncelli, and J.Anthony Movshon. ”Computational Models of Cortical Visual Processing.” Proc.Natl.Acad.Sci.USA Vol.93,pp.623-627 (1996)

[7] Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, Tiejun Huang. ”Optimal ANN-SNN Conversion for High Accuracy and Ultra-Low-Latency Spiking Neural Networks” ICLR (2022)

[8] Dengyu Wu, Xinping Yi and Xiaowei Huang. ”A Little Energy Goes a Long Way: Build an Energy-Efficient, Accurate Spiking Neural Network From Convolutional Neural Network.” Front. Neurosci. (2022)

[9] Jun Haeng Lee, Tobi Delbruck, Michael Pfeiffer, et al. ”Training Deep Spiking Neural Networks using Backpropagation.” Frontiers in neuroscience (2016)

[10] Anatomy of Photoreceptor.cell of a retina in the eye.

[11] Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura et al. ” A Primer on Zeroth-Order Optimization in Signal Processing and Machine Learning.” IEEE Signal Processing Magazine. (2020)

[12] Nguyen-Dong Ho, Ik-Joon Chang. ”TCL: an ANN-to-SNN Conversion with Trainable Clipping Layers.” ACM/IEEE Design Automation Conference(DAC) (2021)

[13] RQ Quiroga, L Reddy, G Kreiman, C Koch and I Fried ”Invariant visual representation by single neurons in the human brain.” Nature, (2005). 435(7045), 1102-1107.

Disclaimer

Since I have no previous experience in essay writing, I used LLM for language polishing in a part of my paper, but did not use LLM to generate any research content, and I have carefully checked the polished content.