Scaling SNNs Trained Using Equilibrium Propagation to Convolutional Architectures

Jiaqi Lin, Malyaban Bal, Abhronil Sengupta
School of Electrical Engineering and Computer Science
The Pennsylvania State University
University Park, PA 16802, USA
Email: {jkl6467, mjb7906, sengupta}@psu.edu

***Abstract*—Equilibrium Propagation (EP) is a biologically plausible local learning algorithm initially developed for convergent recurrent neural networks (RNNs), where weight updates rely solely on the connecting neuron states across two phases. The gradient calculations in EP have been shown to approximate the gradients computed by Backpropagation Through Time (BPTT) when an infinitesimally small nudge factor is used. This property makes EP a powerful candidate for training Spiking Neural Networks (SNNs), which are commonly trained by BPTT. However, in the spiking domain, previous studies on EP have been limited to architectures involving few linear layers. In this work, for the first time we provide a formulation for training convolutional spiking convergent RNNs using EP, bridging the gap between spiking and non-spiking convergent RNNs. We demonstrate that for spiking convergent RNNs, there is a mismatch in the maximum pooling and its inverse operation, leading to inaccurate gradient estimation in EP. Substituting this with average pooling resolves this issue and enables accurate gradient estimation for spiking convergent RNNs. We also highlight the memory efficiency of EP compared to BPTT. In the regime of SNNs trained by EP, our experimental results indicate state-of-the-art performance on the MNIST and FashionMNIST datasets, with test errors of 0.97% and 8.89%, respectively. These results are comparable to those of convergent RNNs and SNNs trained by BPTT. These findings underscore EP as an optimal choice for on-chip training and a biologically-plausible method for computing error gradients.**

***Index Terms*—Spiking Neural Networks, Equilibrium Propagation, Local Learning.**

I. Introduction

Equilibrium Propagation (EP) has surfaced as an efficient and biologically plausible learning framework for training energy-based neural architectures, namely convergent recurrent neural networks consisting of bidirectionally connected neurons [1]. It offers an unified computational circuit during two phases of training through the minimization of an energy function. In contrast, Backpropagation (BP) separates the computational circuits into the forward and backward passes. In backward pass, the explicit calculation of error signal requires a circuit different from the forward pass, and is considered to be biologically implausible [2]. In EP, error signals are propagated from local perturbations on the last layer of neurons through backward connection to preceding layers. The implicit computation on errors features spatial locality that benefits neuromorphic computations [3]. Theoretical analysis of EP [4] has been shown to approximate Backpropagation Through Time (BPTT) [5], [6], and the weight updates of EP resembles Spike-Timing-Dependent Plasticity (STDP) [1], [7], which advocates EP as an outstanding biologically plausible alternative to BPTT. However, EP suffers from the first-order bias in the gradient estimate due to finite nudging. Earlier works on EP, therefore were restricted to shallow fully-connected architectures [1], [8]–[10]. To overcome this limitation, researchers incorporated both positive and negative nudge phases to cancel out the bias introduced, subsequently allowing for a more accurate gradient estimation. Recent advancements [11], [12] have reduced the accuracy disparity between EP and BP on both vision-based datasets and natural language processing based sequential learning tasks.

Spiking Neural Networks (SNNs) are considered to be a more biologically plausible neural model where the information communicated between neurons depends on spiking events. The event-driven mechanism significantly reduces the energy consumption of neural networks due to the sparsity of neuron activities [13]–[15]. The majority of current works for training SNNs from scratch uses BP based methods [16]–[21], which reduces the biological significance of the former. EP leverages unified circuitry and inherent dynamics of neural systems to adjust synaptic weights, which is analogous to how learning might occur in biological systems. The biological plausibility of Equilibrium Propagation (EP), coupled with its recent strides in narrowing accuracy gaps, indicates its potential to serve as a viable alternative for training SNNs. A recent study proposed a formulation of EP with a step-size scheduler that adjusts the accumulation rate of neurons states, allowing SNNs to converge to the same stable states as convergent RNNs [8]. However, there is a performance gap between this work and state-of-art performance on the MNIST dataset. A simpler yet effective methodology of training SNNs using EP has been also proposed with the only requirement of leaky integrate-and-fire (LIF) neurons [3] in the network formulation. Nonetheless, this simple EP system experiences scalability issues. Experimentally, extending this formulation into multiple linear layers results in vital performance degradation. Both studies on SNNs are confined to linear layers and only the MNIST dataset. Given the effectiveness of EP in training continuous-valued convolutional neural networks [9], [11], it is imperative to develop an approach to train convolutional spiking convergent RNNs using EP. However, convolutional architecture formulations from the continuous-

valued domain cannot be directly applied to spiking convergent RNNs and therefore needs significant rethinking in architectural modules like maximum pooling and its inverse, namely unpooling operations. The primary contributions of this work are the following:

- We explore for the first time how convolutional spiking convergent RNNs can be trained using EP, bridging the gap between spiking and non-spiking convergent RNNs in the regime of EP.
- We conduct theoretical and experimental analysis on the maximum pooling and unpooling operations to show the information misalignment between forward connections and backward connections. This issue is unique to spiking convergent RNNs trained by EP, which significantly degrades the performance of EP.
- We propose to implement EP by replacing maximum pooling and unpooling with average pooling and nearest neighbor upsampling, which solves the aforementioned problem. The use of average pooling along with upsampling enables us to achieve 0.97% and 8.89% test error on MNIST and FashionMNIST datasets, respectively.

II. Methods

In this section, we first formulate EP by introducing the Hopfield network in terms of convergent RNNs. Then, we revisit methods to quantize EP based on spiking events such that spiking events match the neuron states in convergent RNNs. We subsequently propose convolutional operations in spiking convergent RNNs using EP, and further investigate the information loss in maximum pooling and unpooling operations in the spiking domain.

A. Hopfield Network

A continuous-valued Hopfield Network is a network composed of recurrently connected neurons featuring symmetrical weights ($w_{ij} = w_{ji}$). The energy function $E(\cdot)$ for this type of network is defined as follows [22]:

$$E(s) = \frac{1}{2}\sum_i \xi_i^2 - \frac{1}{2}\sum_{i\neq j} w_{ij}\rho(\xi_i)\rho(\xi_j) - \sum_i b_i\rho(\xi_i) \quad (1)$$

where, ξ_i is the state of neuron i, w_{ij} and b_i are weights and biases of the neuron, and $\rho(\cdot) = [\cdot]_0^1$ is a nonlinear hard sigmoid function that clips the activation of neurons in a range between 0 and 1 [1]. The state transition dynamics of convergent RNNs are derived from this energy function. This imposes an inherent requirement on convergent RNNs, necessitating symmetric bidirectional connections across layers. In convergent RNNs, the temporal dynamics of the neurons is defined as:

$$\frac{\partial \xi^i}{\partial t} = -\frac{\partial E(\xi^i)}{\partial \xi^i} = -\xi^i + \rho'(\xi^i)\left(\sum_j w_{ij}\rho(\xi^j) + b_i\right) \quad (2)$$

Based on the Euler Method, the discretization of this formula gives:

$$\xi_i^t = \rho\left((1-\epsilon)\xi_i^{t-1} + \epsilon\rho'(\xi_i^{t-1})\left(\sum_j w_{ij}\rho(\xi_j^{t-1}) + b_i\right)\right) \quad (3)$$

where, ϵ is the step-size for updating the neuron states. These dynamics allows the neuron to converge to a stable state.

B. Equilibrium Propagation

EP, a two-phase learning algorithm, was initially proposed for training continuous-valued Hopfield Networks [1] and later extended to implement gradient descent in convergent RNNs on a loss function $L(\xi^{\text{out}}, y)$ between some target y and output activation ξ^{out}. In the first phase, named as the free phase, a static input x is fed into the neural network over T_{free} time steps until the state of the network is saturated to a state ξ^* with minimized energy $E(\xi^*)$. The second phase is a nudge phase with a nudging term $\beta\frac{\partial L(\xi^{\text{out}}, y)}{\partial \xi}$, where the parameter β is the nudging factor. The nudging term allows the states of the neural network to settle to a stable state ξ^β in T_{nudge} time steps. The parameter updates are based on the divergence of stable states in the two phases. With $\beta \to 0$, the gradients of EP calculated below approximate the gradient calculated by BPTT [4]:

$$\Delta w = \frac{1}{\beta}\left(\frac{\partial E(\xi^\beta)}{\partial w} - \frac{\partial E(\xi^*)}{\partial w}\right) \quad (4)$$

Laborieux *et al.* observed the estimator bias introduced in the nudge phase with a positive value of β, leading to an inaccurate estimate of gradients [11]. To address this issue, a third phase with a nudging factor of $-\beta$ is introduced such that the estimation biases in the second phase and third phase are canceled out. Equation 4 therefore becomes:

$$\Delta w = \frac{1}{2\beta}\left(\frac{\partial E(\xi^\beta)}{\partial w} - \frac{\partial E(\xi^{-\beta})}{\partial w}\right) \quad (5)$$

C. Spiking Module

Following the energy-based neuron dynamics in Equation 5, O'Connor *et al.* proposed a method that allows neurons with binary communication to converge to the same fixed point as convergent RNNs using EP [8]. In this formulation, the Sigma-Delta modulation $\delta(\cdot)$ [23] mimics the subtractive LIF mechanism of spiking neurons, which is defined as:

$$\begin{aligned} s^t &= \sigma(V^{t-1} + x^t > V_{th}) \\ V^t &= V^{t-1} + x^t - s^t \end{aligned} \quad (6)$$

where, at time step t, V^t is the membrane potential, x^t is the input, s_i^t is the spike generated at time t, and $\sigma(\cdot)$ is the activation function that generates a spike when the membrane potential exceeds the threshold V_{th}. To accelerate binary communication between neurons, predictive coding is exploited to encode and decode real-valued information into bit-streams [8]. In this scheme, to leverage the bandwidth of communication, only temporal changes in neuron states

is sent through binary bit-streams to succeeding neurons via predictive encoder. Subsequently, the predictive decoder receive these changes and estimates current signal valued d_i^t as a linear function of previous samples d_i^{t-1}, where λ is the prediction factor:

$$d_i^t = (1-\lambda)d_i^{t-1} + \lambda \sum_j w_{ij} s_j^{t-1} \tag{7}$$

The predictive encoder quantizes temporal changes in neuron states into binary information, where e_i^t is predicted encoding signal accumulated until time step t:

$$\begin{aligned} e_i^t &= \frac{1}{\lambda}(\rho(\xi_i^t) - (1-\lambda)\rho(\xi_i^{t-1}) \\ s_i^t &= \delta(e_i^t) \end{aligned} \tag{8}$$

D. Spiking Convolutional Layer

Motivated from convolutional architectures for non-spiking convergent RNNs with static input [9], [11], in this paper we propose spiking convolutional layer in the context of energy-based systems. Given N_c number of convolutional layers followed by N_l number of linear layers, input function $\phi(s,t,n)$ sums up weighted quantized inputs of layer n at time t accommodating spike-based communications:

$$\phi(s,t,n) = \begin{cases} \mathcal{P}(w_n * s_{n-1}^{t-1}) + w_{n+1} \tilde{*} \mathcal{P}^{-1}(s_{n+1}^{t-1}), \\ \qquad \text{if} \quad 1 \le n \le N_\text{c} \\ w_n \cdot s_{n-1}^{t-1} + w_{n+1}^T \cdot s_{n+1}^{t-1}, \\ \qquad \text{if} \quad N_\text{c} < n \le N_\text{t} \end{cases} \tag{9}$$

where, $N_\text{t} = N_\text{c} + N_\text{l}$, $\mathcal{P}(\cdot)$ is the pooling operation, $\mathcal{P}^{-1}(\cdot)$ is the inverse of corresponding pooling operation, w_n is the parameter of layer n. To distinguish the operations in linear layers and convolutional layers, let $*$ be the convolution operation, $\tilde{*}$ be the convolution transpose operation, and $\cdot$ be the linear mapping operation. Connecting to Sigma-Delta modulation and predictive coding, the neuron dynamics is formulated as:

$$\begin{aligned} d_n^t &= (1-\lambda)d_n^{t-1} + \lambda\phi(s,t,n) \\ \xi_n^t &= \rho\left((1-\epsilon)\xi_n^{t-1} + \epsilon\rho'(\xi_n^{t-1})\left(d_n^t + b_n\right)\right) \\ e_n^t &= \frac{1}{\lambda}(\rho(\xi_n^t) - (1-\lambda)\rho(\xi_n^{t-1}) \\ s_n^t &= \sigma(V_n^{t-1} + e_n^t > V_{th}) \\ V_n^t &= V_n^{t-1} + e_n^t - s_n^t \end{aligned} \tag{10}$$

To leverage spatial locality, weight updates are directly derived from the states of the two connecting layers n and $n+1$. Accommodating both linear and convolutional layers, Equation 5 becomes:

$$\Delta w_n = \begin{cases} \frac{1}{2\beta}\left(\mathcal{P}^{-1}(\xi_{n+1}^{\beta}) * \xi_n^{\beta} - \mathcal{P}^{-1}(\xi_{n+1}^{-\beta}) * \xi_n^{-\beta}\right), \\ \qquad \text{if} \quad 1 \le n \le N_\text{c} \\ \frac{1}{2\beta}\left(\xi_{n+1}^{\beta} \cdot \xi_n^{\beta^T} - \xi_{n+1}^{-\beta} \cdot \xi_n^{-\beta^T}\right), \\ \qquad \text{if} \quad N_\text{c} < n \le N_\text{t} \end{cases} \tag{11}$$

E. Rethinking Pooling and Unpooling Operations

In this section, theoretical and experimental analysis is provided to justify the reformulation of pooling and unpooling operations in the energy-based spiking convolutional convergent RNN architecture.

Let us first explain the pooling indices mismatch problem in maximum pooling and unpooling operations. Figure 1 illustrates a toy example of information transmission in EP between two consecutive neurons at time steps t and $t+1$. In forward route, R_1 from ξ_n^t to ξ_{n+1}^t, a pooling operator is applied after the convolution of spikes quantized from ξ_n^t, returning down-sampled continuous values with corresponding indices $I^F(X)$. $I^F(X)$ represents the position of the maximum value in each pooling zone, F is the filter size of maximum pooling operator, and X is the input of the pooling operation. Information output from ξ_{n+1}^t is then combined with $I^F(X)$ and subsequently passed to ξ_n^{t+1} through backward connection R_2. $I^F(X)$ is used as the positional indicator to upsample the information implicitly conveying error signal through the inverse of maximum pooling operation. A maximum pooling operation $\mathcal{P}_\text{max}(\cdot)$ is defined as [9]:

$$\mathcal{P}_\text{max}(X,F)_{c,i,j} = \max_{p,q \in [0,F-1]} \left\{ X_{c,F(i-1)+1+p,F(j-1)+1+q} \right\} \tag{12}$$

where, c is the channel, i,j is the spatial position of input X, and p,q are the spatial indices in the pooling zone. The relative indexing $I^F(X)$, computed by Equation 12 within a pooling zone, is defined as:

$$I^F(X)_{c,i,j} = \underset{p,q \in [0,F-1]}{\arg\max} \left\{ X_{c,F(i-1)+1+p,F(j-1)+1+q} \right\} \tag{13}$$

Consider $\mathcal{P}_\text{max}(X,F)$ gives Y and $I^F(X)$. The inverse of

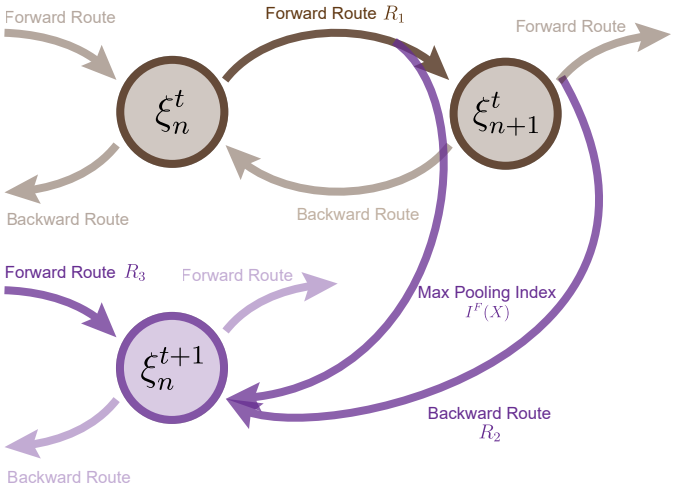

Fig. 1. Information transmission and retention between two consecutive neurons during a period of 2 time steps. Brown color represents information at t, and purple color represents information at $t+1$.

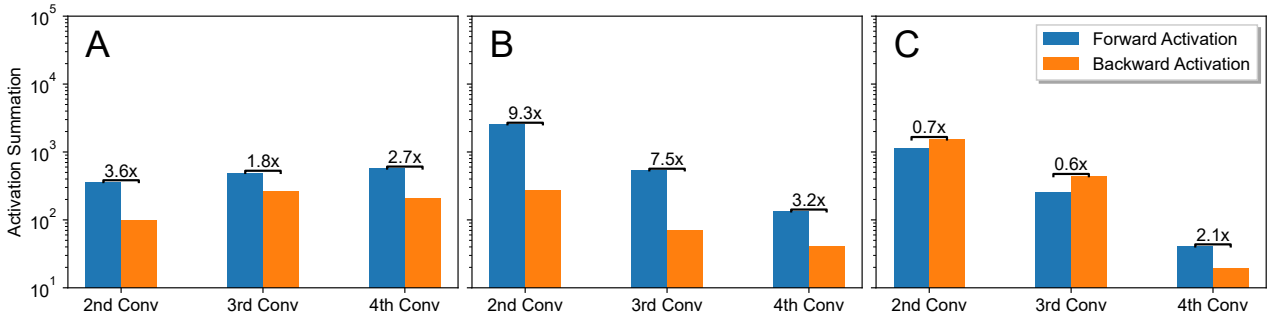

Fig. 2. Activation summation of middle 3 convolutional layers with 32-64-128 channels in both forward route and backward route during nudge phase of a five-layer convolutional architecture. The activations are mean over 2000 random training samples from the MNIST dataset and summed across spatial dimensions of the convolutional layers. (A) Activation summation of convergent non-spiking RNNs equipped with maximum pooling and unpooling operations; (B) Activation summation of SNNs equipped with maximum pooling and unpooling operations; (C) Activation summation of SNNs equipped with average pooling and its inverse operator.

maximum pooling operating on Y is defined as:

$$\mathcal{P}^{-1}_{\text{max}}(Y, I^F(X))_{c,i,j} = \begin{cases} Y_{c,\lceil i/F \rceil, \lceil j/F \rceil} & \text{if } \{i,j\} \in I^F(X) \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

In convergent RNNs, Y returned by $\mathcal{P}_{\text{max}}(X, F)$ forms a one-to-one relationship with indices $I^F(X)$ in forward route R_1, where $X = w_n * \xi^t_n$. The information in Y is retained in the evolution of neuron states ξ^t_{n+1}, and is returned in backward route R_2 without the loss of information. However, in the backward route R_2 of the spiking implementation, Y returned by $\mathcal{P}_{\text{max}}(X, F)$ is quantized through Sigma-Delta modulation and predictive coding, as described in Equation 10. During the process of updating neuron states ξ^t_{n+1}, Y is partially quantized into binary spikes s^t_{n+1} and partially accumulated as potential V^t_{n+1}, thereby resulting in a possibility of information mismatch with $I^F(X)$. Consequently, $I^F(X)$ cannot be used as a positional indicator for the inverse of maximum pooling in backward route R_2. An alternative solution is to perform maximum pooling after the quantization step. However, this approach leads to inconsistencies in the spatial dimensions between the forward and backward routes, entailing the use of additional upsampling operators after the inverse of maximum pooling. Specifically, the integration of forward route R_3 and backward route R_2 at neuron state ξ^{t+1}_n necessitates that both inputs have identical spatial dimensions. In the information propagation route, signals at time t pass through two neuron layers, undergoing downsampling twice after the quantization of each neuron state ξ^t_n and ξ^t_{n+1}, and subsequently serve as backward input through R_2 to ξ^{t+1}_n, where signals are upsampled once. Before integration, the forward input does not undergo any downsampling or upsampling in R_3. Consequently, the backward input requires an additional upsampling step to align with the size of the forward input.

Additionally, we show that maximum pooling and unpooling operations creates a significant activation magnitude imbalance between the forward and backward routes in the spiking regime. Experimental results in Figure. 2 presents the activation outputs of the maximum pooling operation and its inverse from both the forward and backward routes during the nudge phase. (The first convolutional layer is excluded since it receives a quantized static image as input in the forward route, making it inappropriate for comparison.) It reveals that magnitude differences in convergent RNNs equipped with maximum pooling and unpooling are less than $4\times$, while a significant gap in activation magnitudes is observed between the forward and backward routes, with differences ranging up to $10\times$ in the spiking regime. This discrepancy is more pronounced in shallower convolutional layers. To investigate why this imbalance is specific to spiking convergent RNNs, consider forward route R_3 and backward route R_2 in Figure. 1, and let $Y_{\text{forward}} = \mathcal{P}(w_n * X_{\text{forward}})$ and $Y_{\text{backward}} = w_{n+1} \tilde{*} \mathcal{P}^{-1}(X_{\text{backward}})$, where X_{forward} and X_{backward} are the inputs of forward routes R_3 and backward route R_2, respectively. Figure. 3 illustrates an increase of both mean and standard deviation from X_{forward} to Y_{forward} and from X_{backward} to Y_{backward} in the spiking domain. Conversely, continuous-valued convergent RNNs exhibit only slight increment. This significant magnitude difference in the spiking scenario is due to the quantization of continuous-valued neuron states to binary spike values, where binary discretization to 0 and 1 values can significantly deviate from the original information and magnify minor differences. In EP, error signal propagation depends on backward routes. The imbalance problem trivializes the significance of the error signal during the accumulation of potentials, resulting in an ineffective nudge phase update of neuron states.

Applying a maximum pooling and its inverse operation in spiking convergent RNNs therefore hinders the accurate estimation of EP. To circumvent these issues, we propose to replace all the maximum pooling and unpooling operations with average pooling and use nearest neighbor unsampling for the inverse of pooling operation. Average pooling and its inverse does not necessitate indices, which avoids the pooling indices mismatch problem. It comes with the advantages of simpler circuit design in neuromorphic hardware, as storing indices require additional states for information retention, and the transmission necessitates extra resources in addition to spike communication. Additionally, the nearest neighbor

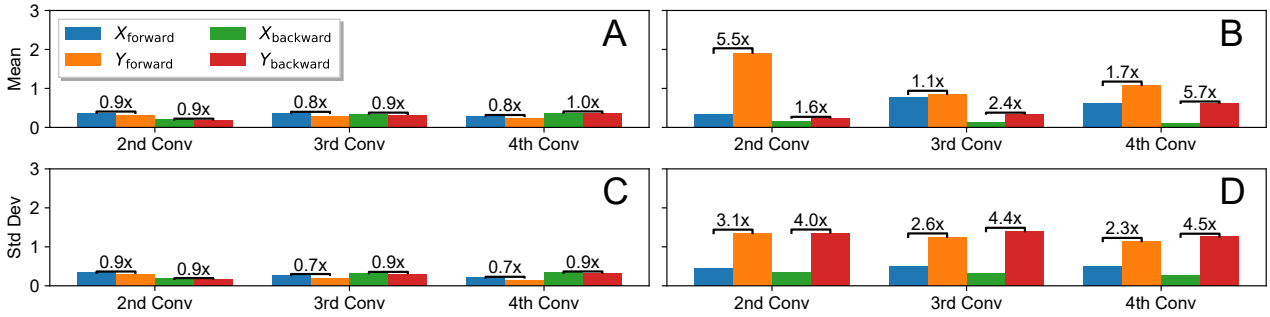

Fig. 3. Mean and standard deviation (Std Dev) of activations X_{forward}, Y_{forward}, X_{backward}, and Y_{backward} of middle 3 convolutional layers with 32-64-128 channels during nudge phase of a five-layer convolutional architecture equipped with maximum pooling and unpooling operations. The average and standard deviation is calculated over 2000 samples from the MNIST dataset. (A) Mean activation of convergent RNNs; (B) Mean activation of SNNs; (C) Standard deviation of convergent RNNs' activations; (D) Standard deviation of SNNs' activations.

upsampling method copies the pixel value from the non-zero input feature map to all positions in a pooling zone, creating a pooling zone without zeros to augment the information passing backwards. This upsampling method mitigates the activation magnitude imbalance problem observed in maximum pooling and its inverse operators. We define the average pooling operations as:

$$\mathcal{P}_{\text{avg}}(X,F)_{c,i,j} = \frac{1}{F^2} \sum_{p,q\in[0,F-1]} X_{c,F(i-1)+1+p,F(j-1)+1+q} \tag{15}$$

We introduce the inverse of average pooling as:

$$\mathcal{P}^{-1}_{\text{avg}}(Y)_{c,i,j} = \frac{Y_{c,\lceil i/F\rceil,\lceil j/F\rceil}}{\alpha} \tag{16}$$

where the scaling factor α controls the magnitude of backward propagating signal. Figure. 2.C presents the activations of average pooling operations from both the forward and backward routes during the nudge phase with $\alpha = F^2$ (chosen as an optimal setting for our experiments and can be tuned further). This demonstrates that average pooling mitigates the magnitude imbalance between the two routes. The experimental performance of our proposed modifications are presented in the following section.

III. Experimental Results

In this section, EP is implemented using the methodology outlined in Section II. Spiking convergent RNNs are utilized for visual classification tasks on the MNIST and FashionMNIST datasets. For comparison, the performance of SNNs trained by BPTT and convergent RNNs trained using both EP and BPTT are also evaluated, denoted as SNN BPTT, CRNN EP, and CRNN BPTT, respectively. With the proposed modifications, we achieve comparable performance on these visual tasks to those of neural networks trained via BPTT. All experiments using BPTT employ the same corresponding architecture as those in EP. Stochastic Gradient Descent (SGD) without momentum is implemented. Additionally, Mean Squared Error (MSE) is applied to spiking convergent RNNs and SNNs, and Cross-Entropy Loss (CE) is applied to convergent RNNs. Optimal hyperparameters are applied to achieve the best performance in each experiment for both EP and BPTT frameworks.

TABLE I
Hyper-parameters for MNIST and FashionMNIST Datasets.

Hyper-parameters	Range	MNIST	FashionMNIST
β	(0.01-1.0)	0.1	0.2
ϵ	(0.1-1.0)	0.9	1.0
λ	(0.1-1.0)	0.6	0.6
T_{free}	(50-500)	250	350
T_{nudge}	(10-100)	50	50
Learning Rate*	(0-1.0)	[0.25, 0.15, 0.1, 0.08]	[0.25, 0.15, 0.1, 0.08]
Batch Size	(10-500)	125	125
Epochs	(10-1000)	250	250

*A layer-wise learning rate is implemented.

The employed convolutional architecture consists of 2 convolutional layers and 2 linear layers and is denoted as 2C2FC. The convolutional layers consists of 128 and 256 channels, respectively, each of which has a kernel size of 3, stride of 1, and padding of 0. Each convolutional layer is followed by a 2-by-2 Average Pooling operation with a stride of two. The weights are initialized using the uniform Kaiming initialization [24]. The range for tuning hyper-parameters is reported in Table I. The experiments were run on one Nvidia RTX 2080 Ti GPU with 11GB memory. Memory consumption is averaged across all batches within a single epoch, with the batch size fixed at 10. Memory consumption of BPTT and EP is averaged over 150 time steps. In EP, free phase, positive nudge phase, and negative nudge phase have 100, 25, and 25 time steps respectively. Training and testing error are averaged over 5 independent trials.

Table II presents the performance of SNNs, and convergent RNNs trained by EP and BPTT. In the table, #FC represents the number of linear layers, #C denotes the number of convolutional layers, and MEM indicates the memory consumption measured in Megabytes. Spiking convergent RNNs trained by EP with the proposed modifications achieve performance at par with SNNs trained by BPTT and convergent RNNs trained by BPTT and EP, while they consume notable reduction in

TABLE II
ERROR AND MEMORY CONSUMPTION OF EP AND BPTT FOR BOTH SPIKING AND NON-SPIKING NETWORKS.

Dataset	Model	SNN EP Error (%)			SNN BPTT Error (%)		CRNN EP Error (%)		CRNN BPTT Error (%)	
		Test	Train	MEM	Test	MEM	Test	MEM	Test	MEM
MNIST	1FC [8]	2.37	0.15	-	-	-	-	-	-	-
	1FC [3]	2.41	1.09	-	-	-	-	-	-	-
	3FC [8]	2.42	0.27	-	-	-	-	-	-	-
	2C1FC [9]	1.02	0.54	-	-	-	-	-	-	-
	2C2FC	0.97 ± 0.32	0.61 ± 0.28	321	0.86 ± 0.11	2171	0.94 ± 0.02	309	0.99 ± 0.36	3018
FashionMNIST	2C2FC	8.89 ± 2.45	2.90 ± 0.02	321	8.71 ± 1.25	2171	8.99 ± 2.89	309	8.83 ± 1.33	3018

memory footprint compared to those trained by BPTT.

IV. DISCUSSION

The EP learning framework regards underlying neural networks as energy-based dynamical systems, providing deeper insights into learning frameworks and serving as a catalyst for further exploration into cognitive processes within the brain. This study is the first exploration of employing EP as a learning mechanism within spiking convergent RNNs incorporating convolution operations, aimed at addressing the scaling challenges of EP to complex visual classification tasks. The performance of EP is limited by indexing mismatch issues caused by maximum pooling and unpooling operations, which are unique to the spiking domain. Substituting maximum pooling and its inverse with average pooling along with nearest neighbor unsampling has proven effective for EP learning. The local weight update feature of EP, both spatially and potentially temporally [10], eliminates the need to store computational graphs for weight updates, significantly reducing the memory consumption during training compared to BPTT. Additionally, it does not rely on surrogate gradients to address the non-differentiable nature of spiking neurons [16], [18], [25]. Coupled with its inherent single-circuit design and STDP-like weight update mechanism [1], EP is emerging as an optimal choice for on-chip training. It not only aligns closely with biological plausibility but also delivers comparable accuracy to iso-architecture ANNs. Future work can explore incorporation of maximum pooling operation in the spiking convergent RNN architecture by overcoming the aforementioned problems along with other innovations in the training process to close the performance gap between EP trained models and state-of-art neural networks trained by BP.

ACKNOWLEDGMENTS

This material is based upon work supported in part by the U.S. National Science Foundation under award No. CAREER #2337646, CCSS #2333881, CCF #1955815, and EFRI BRAID #2318101 and by Oracle Cloud credits and related resources provided by the Oracle for Research program.

REFERENCES

[1] B. Scellier and Y. Bengio, "Equilibrium propagation: Bridging the gap between energy-based models and backpropagation," *Frontiers in computational neuroscience*, vol. 11, p. 24, 2017.

[2] F. Crick, "The recent excitement about neural networks," *Nature*, vol. 337, pp. 129–132, 1989.

[3] E. Martin, M. Ernoult, J. Laydevant, S. Li, D. Querlioz, T. Petrisor, and J. Grollier, "Eqspike: spike-driven equilibrium propagation for neuromorphic implementations," *Iscience*, vol. 24, no. 3, 2021.

[4] B. Scellier and Y. Bengio, "Equivalence of equilibrium propagation and recurrent backpropagation," *Neural computation*, vol. 31, no. 2, pp. 312–329, 2019.

[5] L. B. Almeida, "A learning rule for asynchronous perceptrons with feedback in a combinatorial environment," in *Artificial neural networks: concept learning*, 1990, pp. 102–111.

[6] F. Pineda, "Generalization of back propagation to recurrent and higher order neural networks," in *Neural information processing systems*, 1987.

[7] G.-q. Bi and M.-m. Poo, "Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type," *Journal of neuroscience*, vol. 18, no. 24, pp. 10 464–10 472, 1998.

[8] P. O'Connor, E. Gavves, and M. Welling, "Training a spiking neural network with equilibrium propagation," in *The 22nd international conference on artificial intelligence and statistics*. PMLR, 2019, pp. 1516–1523.

[9] M. Ernoult, J. Grollier, D. Querlioz, Y. Bengio, and B. Scellier, "Updates of equilibrium prop match gradients of backprop through time in an rnn with static input," *Advances in neural information processing systems*, vol. 32, 2019.

[10] ——, "Equilibrium propagation with continual weight updates," *arXiv preprint arXiv:2005.04168*, 2020.

[11] A. Laborieux, M. Ernoult, B. Scellier, Y. Bengio, J. Grollier, and D. Querlioz, "Scaling equilibrium propagation to deep convnets by drastically reducing its gradient estimator bias," *Frontiers in neuroscience*, vol. 15, p. 633674, 2021.

[12] M. Bal and A. Sengupta, "Sequence learning using equilibrium propagation," in *2023 32nd International Joint Conference on Artificial Intelligence (IJCAI), arXiv preprint arXiv:2209.09626*, 2023.

[13] C. Mead, "Neuromorphic electronic systems," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1629–1636, 1990.

[14] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[15] B. V. Benjamin, P. Gao, E. McQuinn, S. Choudhary, A. R. Chandrasekaran, J.-M. Bussat, R. Alvarez-Icaza, J. V. Arthur, P. A. Merolla, and K. Boahen, "Neurogrid: A mixed-analog-digital multichip system for large-scale neural simulations," *Proceedings of the IEEE*, vol. 102, no. 5, pp. 699–716, 2014.

[16] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[17] W. He, Y. Wu, L. Deng, G. Li, H. Wang, Y. Tian, W. Ding, W. Wang, and Y. Xie, "Comparing snns and rnns on neuromorphic vision datasets: Similarities and differences," *Neural Networks*, vol. 132, pp. 108–120, 2020.

[18] B. Yin, F. Corradi, and S. M. Bohté, "Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks," *Nature Machine Intelligence*, vol. 3, no. 10, pp. 905–913, 2021.

[19] M. Bal and A. Sengupta, "Spikingbert: Distilling bert to train spiking language models using implicit differentiation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 10, 2024, pp. 10 998–11 006.

[20] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, "Deep residual learning in spiking neural networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 056–21 069, 2021.

[21] Z. Zhou, Y. Zhu, C. He, Y. Wang, S. YAN, Y. Tian, and L. Yuan, "Spikformer: When spiking neural network meets transformer," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=frE4fUwz_h

[22] J. J. Hopfield, "Neurons with graded response have collective computational properties like those of two-state neurons." *Proceedings of the national academy of sciences*, vol. 81, no. 10, pp. 3088–3092, 1984.

[23] J. C. Candy and G. C. Temes, *Oversampling delta-sigma data converters: theory, design, and simulation*. John Wiley & Sons, 1991.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[25] F. Zenke and T. P. Vogels, "The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks," *Neural computation*, vol. 33, no. 4, pp. 899–925, 2021.