An Efficient non-Backpropagation Method for Training Spiking Neural Networks

Shiqi Guo
Center for Data Science
Peking University
Beijing, China
gsq18@pku.edu.cn

Tong Lin*
Key Laboratory of Machine Perception (MOE)
School of EECS, Peking University
Beijing, China
Peng Cheng Laboratory
Shenzhen, China
lintong@pku.edu.cn

***Abstract*—Spiking Neural Networks (SNNs) have recently attracted significant research interest and have been regarded as the next generation of artificial neural networks due to their suitability for energy-efficient event-driven neuromorphic computing. However, the existing SNNs error backpropagation (BP) method may have severe difficulties in non-differentiable spiking generation functions and vanishing or exploding gradients. In this paper, we introduce an efficient method for training SNNs without backpropagation. The information bottleneck (IB) principle is leveraged to learn synaptic weights and neuron thresholds of an SNN. The membrane potential state for information representation is learned in real-time for higher time and space efficiency compared with the conventional BP method. Experimental results show that the proposed biologically plausible method achieves comparable accuracy and considerable steps/memory reduction in training SNN on MNIST/FashionMNIST datasets.**

***Index Terms*—spiking neural networks, non-backpropagation, membrane potential, information representation**

I. Introduction

The spike neural networks (SNNs), inspired by human brain computing, have recently been regarded as the third generation of artificial neural networks and attracted significant research interest. With the processing of binary spike information, SNNs not only enable higher biological plausibility, but also achieve computational efficiency [1–3]. Moreover, SNNs' are suitable for energy-efficient event-driven processing, such as specialized neuromorphic hardware of IBM's TrueNorth [4] and Intel's Loihi [5], have gained much attention. In order to achieve the same level of performance as conventional deep neural networks, various SNN backpropagation (BP) training methods have been proposed [6, 7]. And the works in [8–13] develop the surrogate derivative to computing the gradients with respect to the spike activation to mitigate the non-differentiability of the spiking function. Recently, the works in [14, 15] further develop the surrogate derivative method for SNN BP training and obtain better experimental results.

However, previous studies of SNN's BP training method still have several disadvantages. The first problem is that the surrogate derivative approaches have limitations in accurate computing gradients, even if a complicated surrogate derivative approach is utilized for the non-differentiability of discrete spike events. Moreover, the obstacle of vanishing gradient and exploding gradient problem due to the gradient chain rule in BP is more significant in the SNN because of the discrete spike events' non-differentiability. Furthermore, the training step is long in the actual SNN implementation. It is not a just-in-time calculation that needs to store historical data with more footprint of memory, which leads to low time and memory efficiency. Last but not least, the error BP method may be considered to have lower biological plausibility, which may be more critical to the more human-like SNN.

To address the above problems in the BP training method, we propose a non-BP training method for SNNs. Our method is inspired by some non-BP training algorithms in the field of conventional neural network training such as classic method attempt [16] or some information-theoretic methods [17, 18]. Our non-BP method not only can mitigate vanishing gradient and exploding gradient but also can have higher biological plausibility based on the information-theoretic IB principle [19]. However, when applying these non-BP training algorithms in [17, 18] to SNNs, it encounters information representation difficulties. To tackle the difficulties, we propose each layer neurons' membrane potential as a feature representation that can partially alleviate the gradient approximation problem. Moreover, it can be realized in real-time, offering higher spatio-temporal efficiency.

This study aims to implement the non-BP training method of SNN to address some problems in the conventional BP training method for SNN. In implementing the method corresponding to the characteristics of SNN, we detect that the membrane potential should be utilized for information representation. The proposed non-BP SNN training method may be a novel attempt for the non-BP training algorithm for SNN within the scope of our knowledge. The non-BP characteristic can mitigate the vanishing gradient and exploding gradient problem and has higher biological plausibility. Moreover, the membrane potential for feature representation characteristics can alleviate the inaccuracy of the surrogate gradient and achieve higher space and time efficiency.

We first implement several different local loss functions to

This work was supported by NSFC Tianyuan Fund for Mathematics (No. 12026606), and National Key RD Program of China (No. 2018AAA0100300).
* Corresponding author

do a comparative experiment to select a better loss function. Then we experiment with the same or deeper network structure of the conventional SNN BP algorithm by selecting the better local loss function. We employ the proposed method on the MNIST [20], and the FashionMNIST [21] datasets. We achieve close to the best results on these datasets, which verifies the feasibility of our non-BP training method of SNN.

II. Background

A. Related work

The related training algorithms for SNNs can be categorized into spike-based error BP and artificial neural network (ANN) to SNN conversion methodologies. The ANN-SNN conversion training methods utilize an ANN to train a corresponding shadow network, which leverages conventional ANN learning methods and converts it to SNN [22–25]. However, such ANN-SNN conversion leads to approximation errors and cannot exploit SNNs' spatio-temporal spiking characteristics.

The well-known SpikeProp algorithm in [6] is one of the earliest BP training methods utilizing a single spike per output neuron for BP. And the work in [7] enables an error BP mechanism for deep SNNs, by treating the membrane potentials of spiking neurons as differentiable signals. Recent works in 2018 [8–10] on BP training methods for SNN utilize the surrogate derivative to mitigate the non-differentiability of the spiking function by computing the gradients with respect to the spike activation. The work in 2019 [13] proposes a neuron normalization technique to adjust the neural selectivity and develop a direct BP learning algorithm for deep SNNs. And the work [14] breaks down error BP across two types of inter-neuron and intra-neuron dependencies to improve temporal learning precision. Moreover, the work in [15] combines the activation-based methods and the timing-based methods for BP training for SNN.

However, the BP training methods have the problems of vanishing gradient, exploding gradient and lower biological plausibility. And some non-BP methods are proposed to address these problems in the field of conventional neural networks. The classic non-BP method in work [16] proposes the dynamic routing between capsules for non-BP training. Furthermore, some information-theoretic non-BP training methods in [17, 18] are proposed based on the IB principle and local loss function. There is no related work on non-BP training methods for SNNs to the best of our knowledge. And we attempt to propose a non-BP training method for SNNs inspired by the non-BP methods for the conventional neural network mentioned above [16–18].

B. Spiking neural network models

We adopt the most common SNN model, named the leaky integrate-and-fire (LIF) neuron model, in this work. The voltage dynamics of N neurons in a single layer can be described as [26]

$$\dot{\mathbf{V}}(t) = -\lambda \mathbf{V}(t) + \mathbf{W}\mathbf{x}(t) + \mathbf{\Omega}\mathbf{s}(t) + \mathbf{I}_{bg}(t), \tag{1}$$

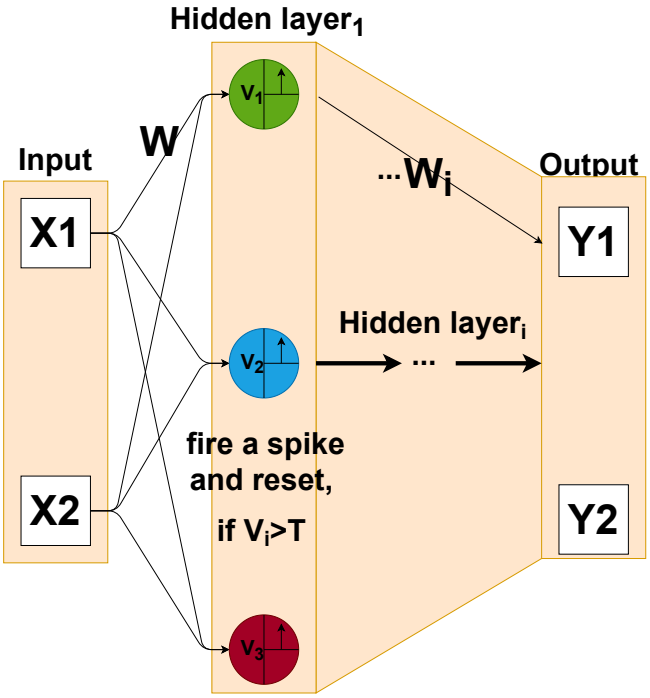

Fig. 1. Illustration of a simple LIF neuron network structure with three hidden neurons in the first hidden layer. The LIF model's neuron fires a spike whenever the neuron's membrane potential crosses its threshold T and then decreases to a low voltage.

where $\mathbf{V}(t) \in \mathbb{R}^N$ is the neurons' membrane potential, λ determines the membrane leak time-constant, $\mathbf{x}(t) \in \mathbb{R}^K$ is the inputs of spike trains or the raw input data, $\mathbf{W} \in \mathbb{R}^{N\times K}$ is the forward weights, $\mathbf{\Omega} \in \mathbb{R}^{N\times N}$ is the recurrent weights, $\mathbf{s}(t) \in \mathbb{R}^N$ is the outputs of post-neural spike trains, and $\mathbf{I}_{bg}(t) \in \mathbb{R}^N$ is background currents or noise. A spike is generated whenever a neuron's membrane potential crosses its threshold T. The N neurons layer's thresholds are $\mathbf{T} \in \mathbb{R}^N$ and the spike generate function is described as a sum of delta-functions

$$\mathbf{s}(t) = \sum_{t_j} \delta\left(t - t_j\right), \tag{2}$$

here t_j is the j-th spiking time when $\mathbf{V}(t_j) > \mathbf{T}$. The recurrent weights $\mathbf{\Omega}$ represent neuron's reset that is a diagonal matrix with all negative diagonal terms. $\mathbf{\Omega}\mathbf{s}(t)$ represents the process of resetting the membrane potential of a neuron after the neuron generates a spike.

A schematic diagram of the LIF neuron network structure is shown in Fig. 1, where the SNN's neurons fire a spike $\mathbf{s}(t) = \delta\left(t - t_j\right)$ and reset membrane potential if $\mathbf{V}(t_j) > \mathbf{T}$. As seen in Fig. 1, the membrane potential of neurons leaks over time when non-neurons spike firing. The LIF model's neuron fires a spike if the neuron's membrane potential crosses its threshold T and reset the neuron's membrane potential. Furthermore, the generated spike forwards to the next layer with weights $\mathbf{W} \in \mathbb{R}^{N\times K}$. To simplify, we fix the background currents or noise $\mathbf{I}_{bg}(t) \in \mathbb{R}^N$ to 0 in this work.

C. Information bottleneck (IB) principle

There are some attempts at non-BP algorithms in the field of conventional neural networks, such as the dynamic routing between capsules [16]. However, we employ the information-theoretic non-BP training methods in [17, 18] based on the IB principle in this work.

The information bottleneck (IB) principle [18, 19] attempts to obtain a trade-off for the hidden representation between the information needed for retaining the input and predicting the output, generalizes the notion of minimal sufficient statistics. The IB objective is formulated as

$$\min_{p_{R_i|X}} I(X;R_i) - \beta I(R_i;Y), \tag{3}$$

where X, Y are the input and label random variables, respectively, and R_i represents hidden feature representation of layer i, $I(X;Y)$ is the mutual information, and β is a trade-off hyperparameter. However, the mutual information in IB is challenging to calculate in practice for several reasons. Based on the IB principle framework, there might be different types of approximation for mutual information in practical applications, so works [17, 18] use different local losses under the IB framework to perform non-BP training for Convolutional Neural Networks(CNNs) [27].

The non-BP training for CNN based on the IB principle framework is schematically shown in Fig. 2. Apply the IB principle in (3) to CNN, and training can be obtained as solving the following optimization problem

$$R_i^* = \arg\min_{R_i} I(R_i;X) - \beta I(R_i;Y), \tag{4}$$

so the local loss function can be expressed as

$$L_i = I(R_i;X) - \beta I(R_i;Y), \tag{5}$$

where L_i is the local loss function on i-th layer. Note that the IB-type global loss function in (3) can be optimized by minimizing the local loss function (5) calculated by the information representation of each layer, respectively. Here the non-BP training method can be realized by minimizing the local loss function separately, as shown in Fig. 2.

Considering the higher biological plausibility, it is possible that there is no BP in the process of human neuron learning. Therefore, it obtains higher biologically plausibility by non-BP training for SNN. Moreover, non-BP training for SNN also mitigates the vanishing gradient and exploding gradient problem. The non-BP training method we proposed in this paper combines the IB framework of section II-C and the LIF SNN model of section II-B.

III. Method

A. Information representation and forward propagation

When we consider applying the information-theoretic IB principle framework in Section II-C to SNN, a problem first arises: what exactly is utilized to represent the information in SNN hidden layers? Suppose we utilize each layer's spike output tensor in the conventional SNN to represent information.

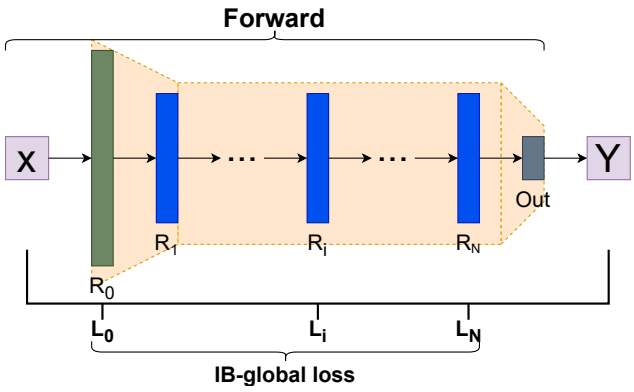

Fig. 2. Illustration of the IB principle framework for non-BP training in (3). The IB-type local loss is iteratively obtained during the forward propagation. So that the minimum IB-type global loss can be obtained to realize the non-propagation training method.

For a simple instance, each layer's conventional spike output tensor can be expressed as a tensor with dimensions (N, C, H, W, T), which represent the batch size, channel numbers, image height, image width, and time steps, respectively. To simplify the discussion, we only consider the spike output tensor with dimensions (H, W, T) per layer; the other two (N, C) dimensions can easily get the same inference. If we adopt (3) to perform the calculation of mutual information type loss function, two difficulties will arise:

- Both the input data and the corresponding output labels can be regarded as the approximate representation of information if we consider the requirements of biological plausibility. However, each layer's spike output tensor is a three-dimensional (H, W, T) sparse 0-1 matrix, which is challenging to find the corresponding meaning of the information representation. For instance, suppose that a specific position (h, w, t) is 1, which means generate a spike in time (t) at position (h,w) from the coding point of view. In that case, comparing with the image data at the position (h, w) where pixel grayscale is (t), we might understand how the spike output tensor represents information. To summarize, it means that if there is a spike-generation 1 at the position (h, w, t) in the three-dimensional spike tensor, and the corresponding two-dimensional information representation tensor is magnitude (t) at the (h, w) position. However, the problem is that the time-domain coding assumption of the spike output tensor could be multi-spike contradictions because spikes may be generated at different times in one image training of a conventional SNN.
- If we temporarily neglect the requirements of higher biological plausibility and how each layer's spike output tensor represents information. Moreover, suppose we directly employ the method in (3) (4) corresponding to [18]. We could obtain mutual information by combining the input data, output labels, and spike output tensor with the size of (H, W, T). When we apply the kernel distance

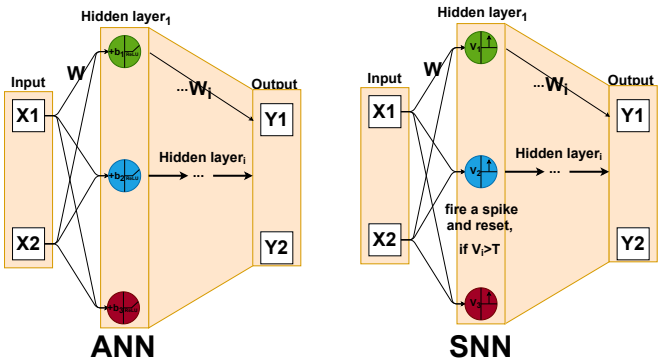

Fig. 3. Illustration of an analogy between the conventional ANN and the SNN.

between features in $k(\mathbf{x}, \mathbf{y}) \sim \exp\left(-\frac{1}{2}\|\mathbf{x}-\mathbf{y}\|^2/\sigma^2\right)$ to calculate the features' kernel distance of the same layer, it will lead to a trivial result that most of the elements are 0 and the rest are sparse constants due to the multi-dimensional sparse 0-1 matrix characteristics of the spike output tensor. Furthermore, the trivial result of the features' kernel distance will lead to the trivial loss function result, which is inappropriate for proposing a non-BP training method for SNN.

To address these difficulties and obtain a suitable information representation in SNN, we compare the layer structure of the conventional ANN with the conventional rectified linear unit (ReLU [28]) activation function and the SNN, as shown in Fig. 3. Each layer of the conventional ANN feature is the output after the activation function

$$ReLU(\mathbf{W}\mathbf{x}+b) = max(\mathbf{W}\mathbf{x}+b, 0), \tag{6}$$

where b is the bias, and the corresponding output of each SNN's layer after the threshold is

$$SO(\mathbf{W}\mathbf{x}(t)) = \begin{cases} 1 & \mathbf{W}\mathbf{x}(t) > \mathbf{T} \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

where SO represents the spike generation process of SNN. The result of this SNN process is a sparse 0-1 tensor with one more time dimension, which has two difficulties discussed above when referring to information representation.

We recognize that the comparison between ANN and SNN in Fig. 3 is similar to comparing the combinational logic circuit [29] and the mealy machine [30] in the sequential logic circuit [31]. As shown in Fig. 4, the mealy machine as a finite state machine [32] has the equivalent expressive potential compared to a combinational logic circuit, which inspires us to utilize the state corresponding to the membrane potential of the SNN as the possibility of representing information [33].

Furthermore, we observe that the threshold vector $\mathbf{T}$ in SNN and the bias in ANN are similar by comparing (6) (7). However, the threshold $\mathbf{T}$ in conventional SNN is preset to a constant value and does not participate in training. This setting will cause the membrane potential of SNN neurons to be limited by a uniform threshold without the meaning of

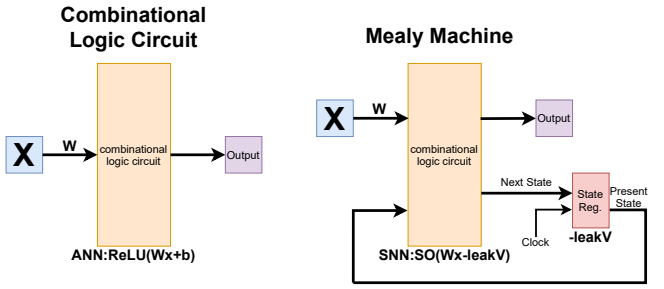

Fig. 4. Illustration of the combinational logic circuit and the mealy machine.

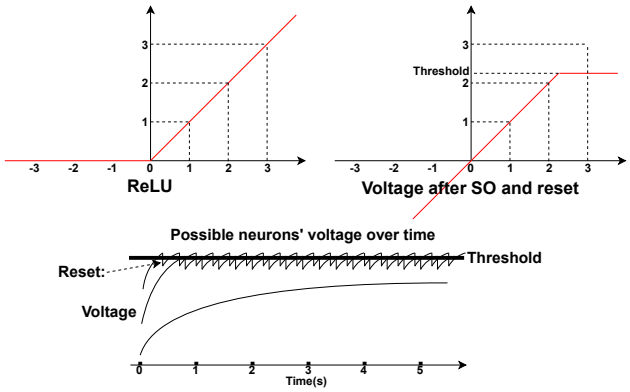

Fig. 5. Illustration of the $ReLU$ function in the ANN and the corresponding membrane potential state after the SO function with the trained threshold in SNN.

information representation. So we analogize the bias in ANN to the SNN threshold $\mathbf{T}$, which can also be trained. Then the membrane potential of neurons with various thresholds dynamically represents the information.

We can see a certain degree of symmetry to confirm the credibility of this method when comparing with the $ReLU$ function in the ANN and the corresponding membrane potential state after the SO function with the trained threshold, as shown in Fig. 5. Here we reset the spike-generating neuron's membrane potential to the threshold neighbourhood by modifying the pre-defined parameter $\mathbf{\Omega}$. So the neuron's membrane potential obtains a steady-state oscillation and has a certain symmetry with the ReLU activation function. And utilizing membrane potential for information representation may also obtain higher biological plausibility. For instance, from the perspective of biological plausibility, the spikes between human brain neurons may be an efficient information transmission. The state of the entire brain neuron may be more indicative of information, which can be understood as the neuron's membrane potential in this SNN structure.

Therefore, we propose to utilize the membrane potential to represent the information in the SNN. The forward structure of the entire spiking neural network is an extension of the single-layer structure described in section II-B, introduced neuron threshold learning mechanism within the layer and proper preset parameter $\mathbf{\Omega}$ for achieving the steady-state potential

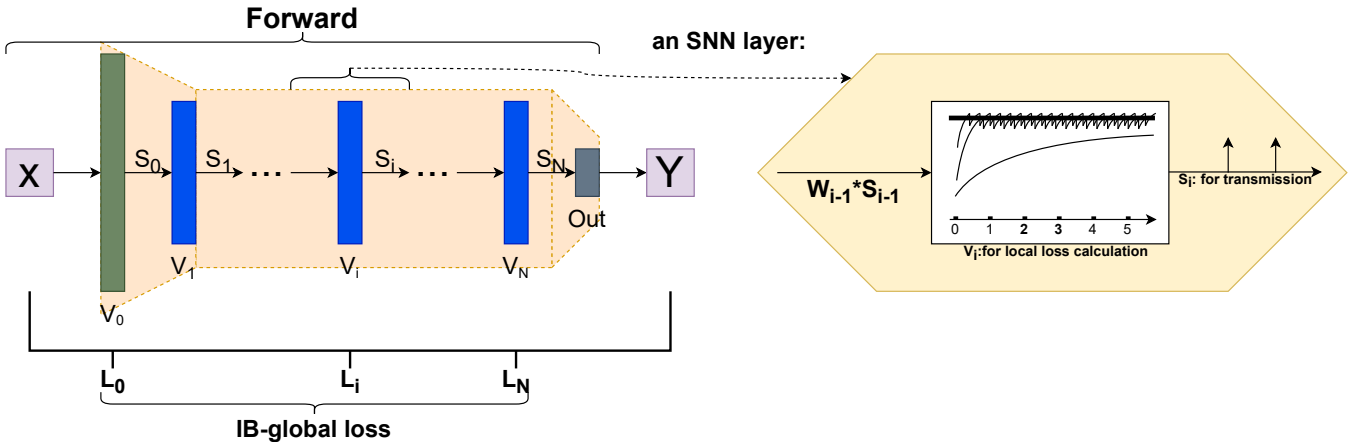

Fig. 6. Illustration of the whole architecture for training SNN without BP, where S_i is the i-th layer's output after SO function, V_i is the i-th layer's neuron membrane potential for information representation. The overall non-BP training network structure for SNN is based on the IB principle framework in Fig. 2. The output of each SNN layer is bisected, where the SNN neurons' membrane potential is participated in the calculation of the local loss function to train the network parameters, and the spikes output from each layer after SO is forwarded to the next layer for information transmission.

oscillation.

From equation (1), the forward propagation process of each SNN layer can be expressed as

$$\dot{\mathbf{V}}(t) = -\lambda \mathbf{V}(t) + \mathbf{W}\mathbf{x}(t) + \mathbf{\Omega}\mathbf{s}(t), \tag{8}$$

here we preset the background noise $\mathbf{I}_{bg}(t)$ to 0. Moreover, the next layer's input in the forward propagation is

$$SO(\mathbf{V}(t)) = \begin{cases} 1 & \mathbf{V}(t) > \mathbf{T} \\ 0 & \text{otherwise} \end{cases}. \tag{9}$$

B. Loss function and overall network structure

We can obtain the overall network structure by utilizing the forward propagation SNN structure described in section III-A combined with the IB principle in section II-C. The schematic description of the entire network structure is shown in Fig. 6.

Among the whole SNN structure, the membrane potential of each SNN layer can be regarded as the feature for information representation and participates in calculating local loss function. Applying (5) to the i-th layer membrane potential, we get the IB objective (IB-type local loss) in SNN

$$L_i = I\left(X; V_i\right) - \beta I\left(V_i; Y\right), \tag{10}$$

where V_i is the i-th layer's neuron membrane potential like the features in CNN. However, the mutual information in (10) may be difficult to calculate for various reasons. We can obtain several types of local loss by the different mutual information approximations in the IB principle framework.

We attempt four different local loss functions of different mutual information approximations in our experiment. We compare the results of different local loss functions to select the local loss function that realizes better performance in SNN to continue further experiments. For example, we examine a distance projection as local loss function, refer to [26]. The distance projection function L_{proj} is

$$L_{proj} = \frac{1}{2}\sum_i \left\| \mathbf{P}_i^{\top}(Y - \tilde{Y}) \right\|^2, \tag{11}$$

where $\mathbf{P}_i$ is the i-th projection matrix, Y, $\tilde{Y}$ are the data labels and predicted data labels, respectively.

We further examine the cross-entropy and the feature-correlation as local loss function, and the combination of these two local loss functions [17]. The cross-entropy function L_{ce} is

$$L_{ce} = CrossEntropy\left(Y, D^T V_i\right), \tag{12}$$

where D^T is the decoder matrix for decoding local layer features V_i to $\tilde{Y}$. The feature-correlation function L_{corr} is

$$L_{corr} = \|C(V_i) - C(Y)\|_F^2, \tag{13}$$

where $C(X)$ is the correlation matrix in which an element c_{ij} is

$$c_{ij} = c_{ji} = \frac{\widetilde{\mathbf{x}}_i^T \widetilde{\mathbf{x}}_j}{\|\widetilde{\mathbf{x}}_i\|_2 \|\widetilde{\mathbf{x}}_j\|_2}. \tag{14}$$

Combining the above two local loss functions with a trade-off, the local loss function $L_{\text{ce-corr}}$ is,

$$L_{\text{ce-corr}} = (1-\beta)L_{ce} + \beta L_{corr}, \tag{15}$$

where β is the weight parameter.

And for the different local loss functions of mutual information approximations represented by membrane potential V_i, the unified local loss gradient can be calculated by

$$\frac{\partial L}{\partial T_i} = \frac{\partial L}{\partial V_i}\frac{\partial V_i}{\partial T_i}, \tag{16}$$

$$\frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial V_i}\frac{\partial V_i}{\partial W_i} = \frac{\partial L}{\partial V_i}\mathbf{x}(t), \tag{17}$$

where T_i is the i-th layer neurons' threshold. When we use membrane potential as information representation, the gradient of V_i versus W_i can be easily obtained, and the gradient of V_i versus T_i can be expressed as

$$\begin{aligned}\frac{\partial V_i}{\partial T_i} &= \frac{\partial}{\partial T_i}(T_i - V_i)H(V_i - T_i) \\ &= (T_i - V_i)\delta\left(V_i - T_i\right) + 1 \approx \alpha_1 \mathbf{s}(t) + \alpha_2,\end{aligned} \tag{18}$$

where $H(x)$ is the Heaviside step function and $\delta(x)$ is the Dirac delta function. And we approximate the Dirac-function in the implementation by the hyperparameters α_1, α_2, ($\alpha_1 \gg \alpha_2$). Moreover, the $(T_i - V_i)H(V_i - T_i)$ is the expression with T_i for $\mathbf{\Omega}\mathbf{s}(t)$ in (8), in order to properly preset parameter $\mathbf{\Omega}$ for achieving the steady-state potential oscillation.

To summarize, we propose the whole non-BP training method for SNN by integrating the SNN forward-propagation (8), the IB principle framework (10), and the single-layer local loss gradient (16), (17), (18). The non-BP training method not only has higher biological plausibility, but also can mitigate the vanishing gradient and exploding gradient problem.

Furthermore, the spike generation function $\mathbf{s}(t)$ does not appear in the local loss function, which solves the non-differentiable spike generation function problem and alleviates the surrogate gradient's inaccuracy. In other words, the output of each SNN layer is bisected, the membrane potential participates in the calculation of the local loss function to train the network parameters, and each layer spike output is forwarded to the next layer for information transmission. And the membrane potential as an information representation is similar to the state register in our SNN training method. Therefore, the proposed method can calculate the current state potential V in real-time instead of storing one more time-dimension tensor data like the conventional SNN method, achieving higher memory efficiency. Also, since the stable oscillation state of the membrane potential V can be obtained quicker by real-time calculation, it has fewer time steps and higher time efficiency than the conventional SNN method. Overall, the membrane potential for feature representation characteristics can alleviate the inaccuracy of the surrogate gradient and achieve higher memory and time efficiency.

IV. Experiments

We first experiment with the proposed non-BP training method with four different local loss functions in Section III-B, which is the experiment IV-A with the multilayer perceptron (MLP) network structure on the MNIST dataset. Then we compare and select the local loss functions with the best experimental results for further experiments. Further experiments on the non-BP training method apply the equivalent or the deeper network structure as the conventional SNN on MNIST/FashionMNIST and obtain comparable accuracy results. Furthermore, the proposed non-BP training method can mitigate the vanishing gradient and exploding gradient problem that could train deeper networks and achieve higher memory and time efficiency. The details of our implementation and experimental settings are available at https://github.com/worden1/ICTAI_snn.

TABLE I
Performances of non-BP SNNs on MNIST with different losses

Local loss	Type	Network	Epochs	Accuracy
L_{proj}	non-BP	100-100	100	33.47%
L_{ce}	non-BP	1024-1024-1024	100	93.89%
L_{corr}	non-BP	1024-1024-1024	100	90.96%
$L_{ce-corr}$	non-BP	1024-1024-1024	100	91.35%

A. Local loss function comparison

Although SNN has recently attracted significant research interest as a possible third-generation artificial neural network, only several latest works [14, 22] have achieved the same level performance results as ANN on the challenging data set CIFAR10 [34]. However, other latest works are also mainly on some simple data sets like MNIST for experiments. Therefore, we first examine the performance of our proposed non-BP training method using different local loss functions on the MNIST dataset with MLP type SNN structure, and the results are shown in Table I. We can see the experimental results on MNIST of four different local loss functions and discover that the local loss function in (12) offers the best result. It is shown that a better mutual information approximation for local loss function that is more suitable for SNN characteristics might significantly improve the results' accuracy. Therefore, the local loss function L_{ce} is selected for further non-BP method experiments.

B. MNIST

We utilize the loss function L_{ce} in (12) for the further experiments. The engineering implementation of the whole SNN layer is stored as a real-time tensor due to threshold participation training. Moreover, the tensor dimension is (N, C, H, W), which is different from the conventional SNN structure (N, C, H, W, T). The proposed SNN structure is real-time and does not need to store one more tensor dimension (T). Therefore, the memory footprint is diminished, and long-time simulation can be easily performed without worrying about insufficient memory. Besides, it also obtains higher time efficiency by quickly achieving the membrane potential stable oscillation state.

We experiment with the same network structure with the conventional BP method for SNN and a VGG [35] structure on the MNIST data set. We compare the results with some conventional SNN results in Table II. The proposed method real-time nature can reduce memory usage and take fewer time steps because the membrane potential quickly reaches a steady state. From the results of the step/memory reduction column in Table II we can see that our proposed non-BP method can have higher time and memory efficiency, and the corresponding step/memory reduction can reach up to 133.3/400x if we use our method's step/memory occupation as a benchmark 1/1x. Training on the deeper VGG-like structure shows that the

TABLE II
PERFORMANCES OF NON-BP METHOD FOR TRAINING SNNS ON MNIST

Method	Type	Network	Time steps	Steps/Memory reduction	Epochs	Accuracy
HM2BP[10]	BP	15C5-P2-40C5-P2-300	400	133.3/400x	100	99.42%
SLAYER[8]	BP	12C5-P2-64C5-P2	300	100/300x	100	99.36%
Spiking CNN[7]	BP	20C5-P2-50C5-P2-200	>200	66.7/200x	150	99.31%
STBP[9]	BP	15C5-P2-40C5-P2-300	>100	33.3/100x	200	99.42%
TSSL[14]	BP	15C5-P2-40C5-P2-300	5	1.6/5x	100	**99.50**%
ours	non-BP	15C5-P2-40C5-P2-300	3	1/1x	100	98.19±0.19%
ours	non-BP	VGG8A	3	1/1x	100	98.96±0.18%

12C5: convolution layer with 12 of the 5×5 filters. P2: pooling layer with 2×2 filters.

TABLE III
PERFORMANCES OF NON-BP METHOD FOR TRAINING SNNS ON FASHIONMNIST

Method	Type	Network	Time steps	Steps/Memory reduction	Epochs	Accuracy
ANN[21]	BP	32C5-P2-64C5-P2-1024	–	–	100	91.60%
HM2BP[10]	BP	400-400	400	133.3/400x	100	88.99%
TSSL[14]	BP	32C5-P2-64C5-P2-1024	5	1.6/5x	100	**92.83**%
ours	non-BP	32C5-P2-64C5-P2-1024	3	1/1x	100	87.74±0.22%
ours	non-BP	VGG8A	3	1/1x	100	87.92±0.33%

proposed method might mitigate the gradient vanishing and gradient exploding through the non-BP training process. The proposed method achieves close to the best results. However, our method is non-BP comparing with that most of the conventional SNN training methods are BP methods.

C. FashionMNIST

We employ the same network structure with the conventional BP method for SNN and a deeper VGG structure network on the FashionMNIST data set, and the experiment result is listed in Table III. The test accuracy of our method is about 88% which is comparable to competing methods. However, the experimental results slightly have less accuracy than other conventional approaches, perhaps because the hyperparameters or network structure are not optimal. Besides, the results of the step/memory reduction in Table III show that our proposed non-BP method can have higher time and memory efficiency, and the corresponding step/memory reduction can reach up to 133.3/400x. Also, training on the deeper VGG-like structure shows that the proposed method might mitigate the gradient vanishing and gradient exploding problems and obtain higher biological plausibility through the non-BP property.

V. CONCLUSION

We aim to implement a potential-state based method for training spiking neural networks without BP in this work, which may be a novel attempt within the scope of our knowledge. The proposed method calculates the mutual information between hidden features and datasets as the local loss function to perform non-BP training of the SNN parameters, which is essentially based on the IB principle framework. We find that the membrane potential instead of the spike output tensor can be utilized as the information representation in the SNN. Correspondingly the proposed method introduces the threshold to participate in the training and obtains higher biological plausibility.

This proposed non-BP method for training SNN may have the following advantages compared with the existing BP method:

- It mitigates the difficulties of possible gradient vanishing and gradient exploding during the conventional BP process because it does not use a deep gradient chain rule.
- It allows to train deeper SNN for which BP training fails, and it is potentially possible to enable parallel training of each SNN layer because of non-BP chain rules.
- It alleviates the inaccuracy of the surrogate gradient due to the non-differentiability, since the non-differentiable spiking generation function does not appear in the local loss function by applying membrane potential-state for local loss calculation.
- It achieves higher memory and time efficiency, since the membrane potential-state can calculate the local loss in real-time and quickly achieve the stable oscillation state.

However, our method as a preliminary exploration still has some limitations that need further work. For instance, we can experiment on more data sets, improve the accuracy of the result, and continue to find the local loss function more suitable for the characteristics of SNN. Moreover, combined with the real-time characteristics of our proposed method, it may be applied to other application scenarios, such as time series processing or video data processing.

REFERENCES

[1] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[2] W. Gerstner and W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge university press, 2002.

[3] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," *Nature*, vol. 575, no. 7784, pp. 607–617, 2019.

[4] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*,

"TrueNorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-aided design of integrated circuits and systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[5] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[6] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1-4, pp. 17–37, 2002.

[7] J. H. Lee, T. Delbruck, and M. Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in Neuroscience*, vol. 10, p. 508, 2016.

[8] S. B. Shrestha and G. Orchard, "SLAYER: Spike layer error reassignment in time," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[9] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in Neuroscience*, vol. 12, p. 331, 2018.

[10] Y. Jin, W. Zhang, and P. Li, "Hybrid macro/micro level backpropagation for training deep spiking neural networks," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[11] D. Huh and T. J. Sejnowski, "Gradient descent for spiking neural networks," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[12] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[13] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, "Direct training for spiking neural networks: Faster, larger, better," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1311–1318.

[14] W. Zhang and P. Li, "Temporal spike sequence learning via backpropagation for deep spiking neural networks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[15] J. Kim, K. Kim, and J.-J. Kim, "Unifying activation-and timing-based learning rules for spiking neural networks," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[16] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] A. Nøkland and L. H. Eidnes, "Training neural networks with local error signals," in *International Conference on Machine Learning*, 2019, pp. 4839–4850.

[18] W.-D. K. Ma, J. Lewis, and W. B. Kleijn, "The HSIC bottleneck: Deep learning without back-propagation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5085–5092.

[19] N. Tishby, F. C. Pereira, and W. Bialek, "The information bottleneck method," in *Proceeding of the 37-th Annual Allerton Conference on Communication, Control and Computing*, 1999, pp. 368–377.

[20] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[21] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[22] B. Han, G. Srinivasan, and K. Roy, "Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 558–13 567.

[23] P. U. Diehl, G. Zarrella, A. Cassidy, B. U. Pedroni, and E. Neftci, "Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware," in *IEEE International Conference on Rebooting Computing*, 2016.

[24] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: VGG and residual architectures," *Frontiers in Neuroscience*, vol. 13, p. 95, 2019.

[25] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017.

[26] A. Mancoo, S. Keemink, and C. K. Machens, "Understanding spiking networks through convex optimization," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[27] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, 1989.

[28] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315–323.

[29] M. Karnaugh, "The map method for synthesis of combinational logic circuits," *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 72, no. 5, pp. 593–599, 1953.

[30] M. Shahbaz and R. Groz, "Inferring mealy machines," in *International Symposium on Formal Methods*. Springer, 2009, pp. 207–222.

[31] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh, and M. Papaefthymiou, "Precomputation-based sequential logic optimization for low power," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 2, no. 4, pp. 426–436, 1994.

[32] T. S. Chow, "Testing software design modeled by finite-state machines," *IEEE Transactions on Software Engineering*, no. 3, pp. 178–187, 1978.

[33] T. Wu, L. Pan, Q. Yu, and K. C. Tan, "Numerical spiking neural p systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 6, pp. 2443–2457, 2020.

[34] A. Krizhevsky, V. Nair, and G. Hinton, "The CIFAR-10 dataset," *online: http://www. cs. toronto. edu/kriz/cifar. html*, 2014.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.