Enhancing Representation of Spiking Neural Networks via Similarity-Sensitive Contrastive Learning

Yuhan Zhang*, **Xiaode Liu***, **Yuanpei Chen, Weihang Peng, Yufei Guo**†,
Xuhui Huang, Zhe Ma†

Intelligent Science & Technology Academy of CASIC, Beijing, 100144, China
yfguo@pku.edu.cn, mazhe_thu@163.com

Abstract

Spiking neural networks (SNNs) have attracted intensive attention as a promising energy-efficient alternative to conventional artificial neural networks (ANNs) recently, which could transmit information in the form of binary spikes rather than continuous activations thus the multiplication of activation and weight could be replaced by addition to save energy. However, the binary spike representation form will sacrifice the expression performance of SNNs and lead to accuracy degradation compared with ANNs. Considering improving feature representation is beneficial to training an accurate SNN model, this paper focuses on enhancing the feature representation of the SNN. To this end, we establish a similarity-sensitive contrastive learning framework, where SNN could capture significantly more information from its ANN counterpart to improve representation by Mutual Information (MI) maximization with layer-wise sensitivity to similarity. In specific, it enriches the SNN's feature representation by pulling the positive pairs of SNN's and ANN's feature representation of each layer from the same input samples closer together while pushing the negative pairs from different samples further apart. Experimental results show that our method consistently outperforms the current state-of-the-art algorithms on both popular non-spiking static and neuromorphic datasets.

Introduction

Recent developments in deep neural networks (DNNs) have achieved great success in a variety of computer vision tasks including pattern recognition (Simonyan and Zisserman 2014; He et al. 2016), semantic image segmentation (Chen et al. 2018), object detection (Girshick 2015; Ren et al. 2015), and so on. However, the increasing complexity of these full-precision DNN models brings high energy consumption, which makes them difficult to deploy in real-world resource-constrained environments. The spiking neural network (SNN), inspired by how the brain represents and processes information, has become one of the energy-efficient alternatives of conventional DNNs in some specific scenarios (Ren et al. 2023; Guo et al. 2023b,d). It transmits information via 0/1 spikes thus the multiplication of the SNN's activation and weight can be replaced by the addition. Thanks to such an information processing paradigm, SNNs are more power-efficiency compared with their full-precision DNN counterparts. Moreover, the SNN could be implemented more efficiently on some specialized neuromorphic hardwares (Akopyan et al. 2015; Davies et al. 2018) in an event-driven manner, where only there is a spike coming, the SNN will be activated, otherwise, be silent.

Despite the unique binary spike information processing mode endows SNN with the characteristics of energy saving, it also limits SNN representative ability to some extent. Compared with the full-precision feature representation of the ANNs, the binary spike feature representation of the SNNs leads to limited information capacity and then severe accuracy degradation (Guo et al. 2022c, 2023a). To improve the representative capability of SNN, some previous works advocated using the ANN to guide the training of high-precision SNN based on knowledge distillation (Takuya et al. 2021; Xu et al. 2023; Guo et al. 2023c). These works enhance the training of SNN either by minimizing the Kullback-Leibler divergence (KLD) (Takuya et al. 2021) or distance (Xu et al. 2023) of the features or outputs of ANN and SNN. We note that these methods all only use the samples from the joint distribution of the features or outputs of the ANN and the SNN to assist in the training of SNN. Taking KLD for example,

$$D_{\mathrm{KL}}(P_{\mathbf{S}} \mid\mid P_{\mathbf{A}}) = \sum P_{\mathbf{S}}(s) \log \frac{P_{\mathbf{S}}(s)}{P_{\mathbf{A}}(a)}, \tag{1}$$

where $s = f_{\mathrm{SNN}}(x_i), a = f_{\mathrm{ANN}}(x_i)$, and $(s,a) \in P_{\mathbf{SA}}(s,a)$, $P_{\mathbf{SA}}(s,a)$ denotes the joint distribution. The loss function is obtained by the output features of ANN and SNN based on the same sample input. However, there is also rich information existing between the ANN's and SNN's features obtained by independent sample inputs. More specifically, the product of the respective marginal distributions of ANN's and SNN's representation $P_{\mathbf{S}}(s)P_{\mathbf{A}}(a)$ can also be important for the construction of the loss function. Consequently, we advocate maximizing mutual information between SNN's and ANN's features, through the KLD of joint distribution and the product of marginal distributions, to guide the learning of the SNN. Concretely, the representation ability of SNN will be strengthened not only via pulling SNN's and ANN's features from the same input samples but

*These authors contributed equally.
†Corresponding author.

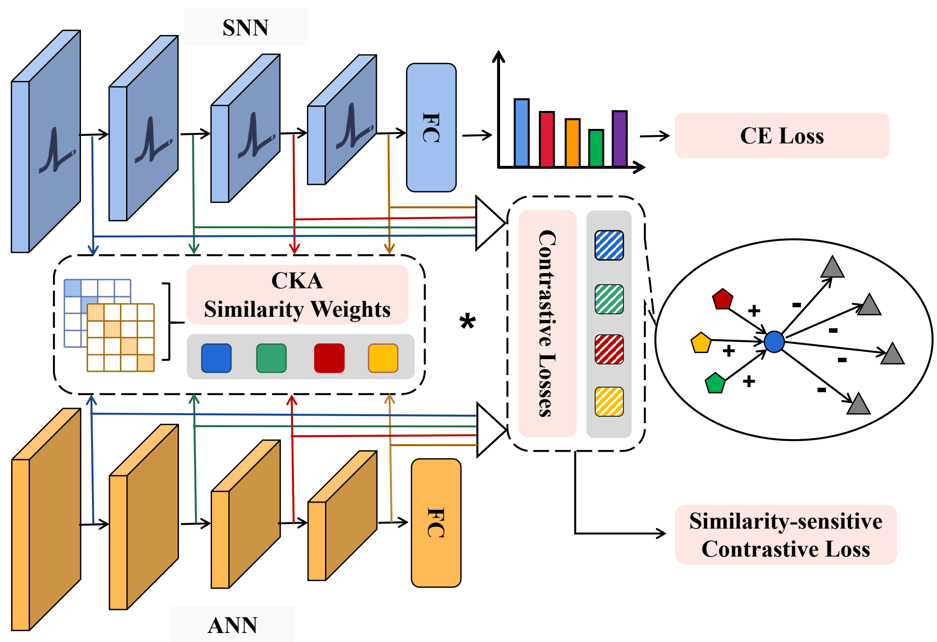

Figure 1: The overall workflow of the proposed method. To enhance the representation of SNN, a similarity-sensitive contrastive loss is introduced for transmitting the important information from ANN's representation to SNN's representation, where contrastive losses at different layers are weighted by CKA.

also by pushing those from different input samples.

In addition, the recent study (Li et al. 2023) has analyzed the similarity of SNN's and ANN's representations, which is measured by centered kernel alignment (CKA). According to the experimental results, we notice that the similarity of SNN's and ANN's representations at different layers is different, and the CKA value shows a decreasing trend with the layers going deep. Considering the difference in representation similarity between the ANN and the SNN along layers, we further introduce the CKA into loss to sense the similarity of the representations and weigh the contrastive loss for mutual information maximization at each layer.

Our training framework is shown in Fig.1. For clarity, the main contribution of this paper is summarized below.

- We propose a novel contrastive learning framework to train SNNs directly via maximizing the mutual information between SNN's representation and its real-valued counterparts of the well-trained ANN. The SNN is optimized based on both joint distribution and the product of marginal distributions of the ANN's and SNN's representation.
- Furthermore, a series of layer-wise similarity indexes are introduced into the total loss to weigh the mutual information maximization at different layers.
- We evaluate our method on both static and neuromorphic datasets. Extensive experimental results under various experimental settings show that our method significantly outperforms state-of-the-art methods, e.g. 78.79% on CIFAR-100, 66.78% on ImageNet, and 80.00% on CIFAR10-DVS.

Related Work

Based on the classification of the review (Guo, Huang, and Ma 2023), the representative ability decreasing of SNNs can be mitigated on the neuron level, network structure level, and training technique level. **On the neuron level**, introducing learnable hyperparameters into the spiking neuron is a common way. LSNN (Bellec et al. 2018) and LTMD (Wang, Cheng, and Lim 2022) proposed adaptive threshold spike neurons to improve the computing and learning capabilities of SNNs. Besides this, introducing a learnable membrane time constant is another commonly used method (Yin, Corradi, and Bohté 2020; Luo et al. 2022). Furthermore, DietSNN (Rathi and Roy 2020a) adopted the learnable membrane leak and firing threshold simultaneously. **On the network structure level**, SEW-ResNet (Fang et al. 2021a) and DS-ResNet (Feng et al. 2022) replaced the standard ResNet backbone with activation before the addition form-based ResNet. The representation capability will be increased due to that this kind of ResNet will fire positive integer spikes. However, the multiplication-addition transformation will be lost at the same time. To handle this problem, MS-ResNet (Hu et al. 2021) proposed a pre-activation form-based ResNet, where the spike-based convolution can be retained. **On the training technique level**, IM-Loss (Guo et al. 2022a) proposed an information maximization loss to improve the activation information entropy. Recently, some works (Takuya et al. 2021; Xu et al. 2023) introduced the distillation method in the SNN domain. In these methods, an ANN teacher is used to guide SNN-student learning with KLD or distance between their outputs or features, aligning ANN's and SNN's output or features obtained from the same sample.

Contrastive learning considers both positive and negative sample pairs to achieve expression learning, which has validated its superiority in many computer vision tasks (Wu et al. 2018; He et al. 2020; Chen et al. 2020; Tian, Krishnan, and Isola 2020, 2019). It pulls the representations of positive sample pairs closer using a contrastive loss function, as well as pushes the representations of positive and negative sample pairs away. This study combines the knowledge distillation method and the contrastive learning method to introduce a robust learning approach with the push-and-pull scheme to effectively enhance the representation of SNNs by the guidance of the ANNs.

Preliminary and Methodology

This section first introduces the spiking neuron model and why its feature's expression ability is limited, then, why and how to align the SNN's representation with the ANN's representation via similarity-sensitive contrastive learning. Finally, the process to train an SNN with the proposed methods will be given in detail by pseudocode.

Spiking Neuron Model

The primary computing neuron of an SNN is much different from that of an ANN. The neuron of an ANN only plays the role of nonlinear transformation and can output real-valued values. While the neuron in an SNN enjoys rich spatially-temporal dynamics. Take the well-known Leaky Integrate-and-Fire (LIF) neuron model as an example. The LIF neuron updates its membrane potential based on the input and its membrane potential at the previous moment as follows,

$$\boldsymbol{U}_{t,\mathrm{pre}} = \tau \boldsymbol{U}_{t-1} + \mathbf{W}\boldsymbol{Z}_t, \tag{2}$$

where $\boldsymbol{U}_{t,\mathrm{pre}}$ denotes the pre-membrane potential at t-th timestep, $\boldsymbol{U}_t$ denotes the membrane potential at t-th timestep, τ is the membrane time constant which represents the leakage effect of the membrane potential, $\mathbf{W}$ is the weight, and $\boldsymbol{Z}_t$ is the binary map comes from the previous layer at t-th timestep.

When $\boldsymbol{U}_{t,\mathrm{pre}}$ exceeds the firing threshold U_{th}, the neuron would fire a spike and reset the U_t to zero, otherwise, the membrane potential would be presented to the next timestep with a leak according to Eq. 2, given by

$$\boldsymbol{O}_t = \begin{cases} 1, \text{ if } \boldsymbol{U}_{t,\mathrm{pre}} \geq U_{\mathrm{th}} \\ 0, \text{ otherwise} \end{cases}, \boldsymbol{U}_t = \boldsymbol{U}_{t,\mathrm{pre}} \cdot (1 - \boldsymbol{O}_t), \tag{3}$$

where U_{th} is a given firing threshold and $\boldsymbol{O}_t$ denotes the output of the LIF neuron. $\boldsymbol{O}_t$ is a binary feature map.

The Representation of SNN

Though the binary spike information processing paradigm is highly energy efficient, it will result in unsatisfactory performance too, since the binary spike activation maps cannot carry enough information compared with ANNs. Take the information entropy concept to analyze it, given a set, $\mathbf{M}$, its representation capability, $\mathcal{R}(\mathbf{M})$ can be measured by the information entropy of $\mathbf{M}$, as follows

$$\mathcal{R}(\mathbf{M}) = \max \mathcal{H}(\mathbf{M}) = \max(-\sum_{m \in \mathbf{M}} p_{\mathbf{M}}(m) \mathrm{log} p_{\mathbf{M}}(m)), \tag{4}$$

where $p_{\mathbf{M}}(m)$ is the probability of a sample, m from $\mathbf{M}$. When $p_{\mathbf{M}}(m_1) = p_{\mathbf{M}}(m_2) = p_{\mathbf{M}}(m_3) \cdots = p_{\mathbf{M}}(m_{N_M})$, $\mathcal{H}(\mathbf{M})$ reaches its maximum, $\log(N_M)$, where N_M is the total number of the samples from $\mathbf{M}$. And for the activation map of the ANN, it can be denoted as $\mathbf{M}_R \in \mathbb{R}^{C \times H \times W}$, where C is the channels and H and W are height and width of the map. Correspondingly, the activation map of the SNN can be denoted as $\mathbf{M}_B \in \mathbb{B}^{T \times C \times H \times W}$, where T is the total timesteps. Since the binary spike output o can be expressed with 1 bit, the number of samples from o is 2. Then, the number of samples from $\mathbf{M}_B$ is $2^{(T \times C \times H \times W)}$ and $\mathcal{R}(\mathbf{M}_B) = \mathrm{log}2^{(T \times C \times H \times W)} = T \times C \times H \times W$. While a real-valued activation for ANN needs 32 bits, thus consisting of 2^{32} samples and $\mathcal{R}(\mathbf{M}_R) = \mathrm{log}2^{32 \times (C \times H \times W)} = 32 \times C \times H \times W$. In most of the direct training SNN works, the T is smaller than 32, thus the representation capability of the SNN is much worse than that of the ANN.

Mutual Information and Centered Kernel Alignment

In this paper, we consider maximizing the mutual information (MI) of the representations of SNN and ANN to align the output features of SNN and ANN, so as to achieve the effect of enhancing the representation ability of the SNN. For ease of notation, we define random variables S^l and A^l for the SNN's and ANN's representation of the data at layer l respectively:

$$\mathbf{S}^l = f_l^S(x), \tag{5}$$

$$\mathbf{A}^l = f_l^A(x). \tag{6}$$

Their mutual information (MI) can be defined as (Solomon 1997):

$$I(\mathbf{S}^l, \mathbf{A}^l) = \sum_{s,a} P_{\mathbf{S}^l \mathbf{A}^l}(s,a) \log \frac{P_{\mathbf{S}^l \mathbf{A}^l}(s,a)}{P_{\mathbf{S}^l}(s) P_{\mathbf{A}^l}(a)}, \tag{7}$$

where $P_{\mathbf{S}^l \mathbf{A}^l}(s,a)$ is the joint distribution, $P_{\mathbf{S}^l}(s) = \sum_a P_{\mathbf{S}^l \mathbf{A}^l}(s,a)$ and $P_{\mathbf{A}^l}(a) = \sum_s P_{\mathbf{S}^l \mathbf{A}^l}(s,a)$ are the marginals of $\mathbf{S}^l$ and $\mathbf{A}^l$, respectively. Compared with (1), mutual information (7) introduces additional information within the product of the respective marginal distributions of ANN's and SNN's representation $P_{\mathbf{S}^l}(s) P_{\mathbf{A}^l}(a)$ to learn the SNN contrastively. It quantifies the amount of information obtained about SNN's representation by observing ANN's representation and can be considered as the reduction in uncertainty about SNN's representation given knowledge of ANN's representation. High mutual information indicates a large reduction in uncertainty and vice versa (Solomon 1997). We would like ANN's and SNN's representations to share as much information as possible, because the more similar they are in the feature space, the smaller the gap between the accuracy of SNN and ANN after the full-connection layer calculation. Theoretically, the mutual information between those two representations should be maximized.

During the training phase, the similarity between SNN and ANN representations of different layers is different (Li et al. 2023), and this difference should be considered and balanced when the mutual information of different layers is integrated in training loss. We uses the trained ANN to guide the training of SNN, which aims at forcing the representation of SNN to be similar to the one of the ANN. Thus we hope our loss can pay more attention to the more different layers. Here we adopt centered kernel alignment (CKA) to sense and then adjust this difference of similarity and introduce it into contrastive loss. CKA computes the similarity between pairs of representation matrices to quantitatively study neural network representations (Kornblith et al. 2019; Cortes, Mohri, and Rostamizadeh 2012).

Given SNN's representation $\mathbf{S}^l \in \mathbb{R}^{n \times T \times p_{s,l}}$ and ANN's representation $\mathbf{A}^l \in \mathbb{R}^{n \times p_{a,l}}$ with the batchsize of n at layer l, the following CKA (Kornblith et al. 2019) is adopted to measure how similar they are.

$$\mathrm{CKA}(\mathbf{K}, \mathbf{L}) = \frac{\mathrm{HSIC}(\mathbf{K}, \mathbf{L})}{\mathrm{HSIC}(\mathbf{K}, \mathbf{K})\mathrm{HSIC}(\mathbf{L}, \mathbf{L})}, \tag{8}$$

where $\mathrm{HSIC}(\mathbf{K}, \mathbf{L}) = \frac{1}{(\mathrm{n}-1)^2}\mathrm{tr}(\mathbf{KHLH})$, where $\mathbf{K} = \mathbf{S}^l\mathbf{S}^{l\top}$, $\mathbf{L} = \mathbf{A}^l\mathbf{A}^{l\top}$ are the Gram matrices with the shape of $n \times n$. $K_{i,j}$ or $L_{i,j}$ implies the similarity between the i-th and j-th example in the representation $\mathbf{S}^l$ or $\mathbf{A}^l$. In order to further measure the similarity between $\mathbf{K}$ and $\mathbf{L}$, we determine whether SNN has an inter-example similarity matrix similar to ANN. Given the centering matrix $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, the Hilbert-Schmidt Independence Criterion (HSIC) proposed by (Gretton et al. 2005) tests the independence of two groups of variables statistically. HSIC= 0 indicates independence. Then, the CKA produces a similarity indicator between 0 and 1 by further normalizing HSIC. The closer CKA is to 1, the more similar the input pairs are. Here, we use the unbiased estimator of HSIC (Song et al. 2012; Nguyen, Raghu, and Kornblith 2020) to calculate it across mini-batches.

$$\mathrm{HSIC}_1(\mathbf{K}, \mathbf{L}) = \frac{1}{n(n-3)} \cdot \left(\mathrm{tr}(\tilde{\mathbf{K}}\tilde{\mathbf{L}}) + \frac{\mathbf{1}^\top\tilde{\mathbf{K}}\mathbf{1}\mathbf{1}^\top\tilde{\mathbf{L}}\mathbf{1}}{(\mathrm{n}-1)(\mathrm{n}-2)} - \frac{2}{\mathrm{n}-2}\mathbf{1}^\top\tilde{\mathbf{K}}\tilde{\mathbf{L}}\mathbf{1}\right). \tag{9}$$

Similarity-Sensitive Contrastive Learning

In this section, we introduce how to construct a similarity-sensitive contrastive loss based on Noise-Contrastive Estimation (NCE) to maximize the mutual information between the layer-wise discrete and the full-precision representations. NCE estimates the mutual information with its lower bound to avoid computing it directly. As shown in Figure 1, the discrete and full-precision representations from the same samples can be pulled close, while representations from different samples can be pushed away, which corresponds to the core idea of contrastive learning. Then, layer-wise NCE losses are weighted by the corresponding CKA scale, thus forming the similarity-sensitive contrastive loss.

For a training batch with N samples, the samples can be denoted as: $\{x_i\}(i \in 1, ..., N)$. Let us define a distribution q with latent variable G which decides whether a contrastive pair $(f^S(x_i), f^A(x_j))$ comes from the same samples ($i = j \Leftrightarrow G = 1$) or different samples ($i \neq j \Leftrightarrow G = 0$):

$$q(\mathbf{S}^l, \mathbf{A}^l | G = 1) = p(\mathbf{S}^l, \mathbf{A}^l), \tag{10}$$

$$q(\mathbf{S}^l, \mathbf{A}^l | G = 0) = p(\mathbf{S}^l)p(\mathbf{A}^l). \tag{11}$$

Suppose that in the data, for every $N-1$ incongruent pair (different samples input to f_l^S and f_l^A from the product of marginal distributions), 1 congruent pair (the same sample input to f_l^S and f_l^A) is given. Then the priors of the latent variable G are:

$$q(G = 1) = \frac{1}{N}, q(G = 0) = \frac{N-1}{N}. \tag{12}$$

Through simple operations and Bayes' rule, the posterior for $G = 1$ can be obtained by:

$$\begin{aligned} &q(G = 1 | \mathbf{S}^l, \mathbf{A}^l) = \\ &\frac{q(\mathbf{S}^l, \mathbf{A}^l | G = 1)q(G = 1)}{q(\mathbf{S}^l, \mathbf{A}^l | G = 0)q(G = 0) + q(\mathbf{S}^l, \mathbf{A}^l | G = 1)q(G = 1)} \\ &= \frac{p(\mathbf{S}^l, \mathbf{A}^l)}{p(\mathbf{S}^l, \mathbf{A}^l) + (N-1)p(\mathbf{S}^l)p(\mathbf{A}^l)}. \end{aligned} \tag{13}$$

Next, taking the logarithm on both sides of the above equation, we have:

$$\begin{aligned} &\log q(G = 1 | \mathbf{S}^l, \mathbf{A}^l) \\ &\quad = \log \frac{p(\mathbf{S}^l, \mathbf{A}^l)}{p(\mathbf{S}^l, \mathbf{A}^l) + (N-1)p(\mathbf{S}^l)p(\mathbf{A}^l)} \\ &\quad = -\log(1 + (N-1)\frac{p(\mathbf{S}^l)p(\mathbf{A}^l)}{p(\mathbf{S}^l, \mathbf{A}^l)}) \\ &\quad \leq -\log(N-1) + \log \frac{p(\mathbf{S}^l, \mathbf{A}^l)}{p(\mathbf{S}^l)p(\mathbf{A}^l)}. \end{aligned} \tag{14}$$

Then taking expectation on both sides w.r.t. $q(\mathbf{S}^l, \mathbf{A}^l | G = 1)$ and rearranging, we obtain:

$$\begin{aligned} I(\mathbf{S}^l; \mathbf{A}^l) \geq &\log(N-1) + \\ &\mathbb{E}_{q(\mathbf{S}^l, \mathbf{A}^l | G=1)} \log q(G = 1 | \mathbf{S}^l, \mathbf{A}^l), \end{aligned} \tag{15}$$

where $I(\mathbf{S}^l; \mathbf{A}^l)$ denotes the mutual information between the SNN's and ANN's representations. Therefore, maximizing the mutual information is equivalent to maximizing its lower bound $\mathbb{E}_{q(\mathbf{S}^l, \mathbf{A}^l | G=1)} \log q(G = 1 | \mathbf{S}^l, \mathbf{A}^l)$ w.r.t. the parameters of the SNN. However, the true distribution $q(G = 1 | \mathbf{S}^l, \mathbf{A}^l)$ is not known, but can be estimated instead by fitting a critic function $h : \{S, A\} \rightarrow [0, 1]$ (Tian, Krishnan, and Isola 2019) to samples from the data distribution $q(\mathbf{S}^l, \mathbf{A}^l | G = 1)$ and $q(\mathbf{S}^l, \mathbf{A}^l | G = 0)$, where S and A represent the domains of the representations. According to the properties of h, $(N-1)\mathbb{E}_{q(\mathbf{S}^l, \mathbf{A}^l | G=0)}[\log(1 -$

$h(\mathbf{S}^l, \mathbf{A}^l))]$ is strictly negative. Thus, by simply adding $(N-1)\mathbb{E}_{q(\mathbf{S}^l,\mathbf{A}^l|G=0)}[\log(1-h(\mathbf{S}^l, \mathbf{A}^l))]$ to the inequality (15), the inequality also holds as previous contrastive learning work (Tian, Krishnan, and Isola 2019).

$$I(\mathbf{S}^l; \mathbf{A}^l) \geq \log(N-1) + \mathbb{E}_{q(\mathbf{S}^l,\mathbf{A}^l|G=1)}[\log h(\mathbf{S}^l, \mathbf{A}^l)] + (N-1)\mathbb{E}_{q(\mathbf{S}^l,\mathbf{A}^l)|G=0)}[\log(1-h(\mathbf{S}^l, \mathbf{A}^l))]. \tag{16}$$

Next, let

$$\mathcal{L}^l_{NCE}(h) = \mathbb{E}_{q(\mathbf{S}^l,\mathbf{A}^l|G=1)}[\log h(\mathbf{S}^l, \mathbf{A}^l)] + (N-1)\mathbb{E}_{q(\mathbf{S}^l,\mathbf{A}^l|G=0)}[\log(1-h(\mathbf{S}^l, \mathbf{A}^l))]. \tag{17}$$

In order to learn the SNN by maximizing the mutual information $I(\mathbf{S}^l; \mathbf{A}^l)$, the final optimization problem becomes

$$f^*_S = \arg\max_{f^S} \max_h \mathcal{L}^l_{NCE}(h) \tag{18}$$

Based on inequality (16), for any h, $f^{S*} = \arg\max_{f^S_l} \mathcal{L}^l_{NCE}(h)$ also optimizes a lower-bound on mutual information. Thus, the above learning problem (18) does not rely on h being optimized perfectly. We design our critic function h for our SNN case based on the critic function (Tian, Krishnan, and Isola 2019):

$$h(\mathbf{S}^l, \mathbf{A}^l) = \frac{e^{g^S(\mathbf{S}^l)' g^A(\mathbf{A}^l)/\tau}}{e^{g^S(\mathbf{S}^l)' g^A(\mathbf{A}^l)/\tau} + \frac{N}{M}} \tag{19}$$

where τ is a temperature constant to control the concentration level, and M is the total number of possible pairs. Since the dimensionality of $\mathbf{S}^l$ and $\mathbf{A}^l$ are different, they need to be linearly transformed to the same dimension by g^S and g^A as well as further normalized by L2 norm before inner product.

Finally, for constructing the similarity-sensitive contrastive loss to enhance representations of SNNs, we weigh the NCE loss $\mathcal{L}^l_{NCE}$ by CKA-based weight $\lambda^l_{CKA} = 1/\mathrm{CKA}(\mathbf{K}, \mathbf{L})$ at l-th layer and combine it with classification loss. Then, the total loss L can be defined as:

$$\mathcal{L}_{Total} = \sum_l \lambda^l_{CKA} \mathcal{L}^l_{NCE} + \mathcal{L}_{CE}, \tag{20}$$

where $\mathcal{L}_{CE}$ is the cross-entropy loss. When the representation similarity between ANN and SNN of a certain layer is relatively smaller, The CKA-based weight λ^l_{CKA} will be larger, and the contrastive loss based on mutual information maximization of this layer is greater, and the representation of this layer will be preferentially optimized. We adopt the spatial-temporal backpropagation (STBP) algorithm (Wu et al. 2019) to train the SNN with our method and the following STE surrogate gradients to solve the non-differentiable firing activity of the spiking neuron as other surrogate gradient (SG) methods (Rathi and Roy 2021; Guo et al. 2022b).

$$\frac{dO}{dU} = \begin{cases} 1, & \text{if } 0 \leq \boldsymbol{U} \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

Experiments

In this section, extensive experiments were conducted to validate the effectiveness of the proposed method adopting widely-used spiking ResNet20 (Rathi and Roy 2020b; Sengupta et al. 2018), VGG16 (Rathi and Roy 2020b), ResNet18 (Fang et al. 2021a), ResNet19 (Zheng et al. 2020), and ResNet34 (Fang et al. 2021a) on both static and neuromorphic datasets including CIFAR-10 (Krizhevsky, Nair, and Hinton 2010), CIFAR-100 (Krizhevsky, Nair, and Hinton 2010), ImageNet (Deng et al. 2009), and CIFAR10-DVS (Li 2017). These networks are typically divided into four stages corresponding to four downsamples of the input feature map. The layer-wise representations used in our implementation consist of the representations after these four stages and the output feature representation after a subsequent global averaging pooling. We used the same architecture of ANN and SNN. We first train an ANN and then use it to guide the learning of the homogenous SNN. The hyper-parameters for LIF neuron including the firing threshold U_{th}, the membrane potential decaying τ, and reset potential U_{reset} were set as 0.5, 0.25, and 0 respectively. For static image datasets, the images were fed into the SNN model directly and encoded to $0/1$ spikes using the first layer as recent works (Zheng et al. 2020; Rathi and Roy 2020b). For the neuromorphic image dataset, the $0/1$ spike format was used directly. For comparison results, we list the mean top-1 accuracy and standard deviation when running three times for each experiment.

Ablation Study

To verify the effectiveness of our method, a series of ablation studies were conducted first, including the studies using spiking ResNet20 architecture with different time steps on the CIFAR-100. Table 3 lists the top-1 accuracy of these models. As can be seen, the test accuracy of our SNN models with similarity-sensitive contrastive learning (SSCL) is consistently higher than that of these vanilla SNN models. Moreover, it can be clearly seen that the proposed contrastive loss and the CKA benefit the overall improvement. Specifically, 2-timestep baseline SNN provides an accuracy of 68.27% on CIFAR-100, while SSCL could help SNN improve its accuracy to 69.81%, which is a huge improvement in the SNN field (close to 2.0%).

Comparison with SoTA Methods

We then conducted various experiments on both static and neuromorphic datasets. The details for the datasets and settings are given in the appendix.

CIFAR-10. The result for CIFAR-10 is shown in Table 1. Our models provide better performance than other SoTA methods using three commonly used networks with fewer time steps. Our VGG16 model with only 2 timesteps outperforms the KDSNN (Xu et al. 2023) with 4 timesteps by 2.78% accuracy. This demonstrates that, compared with the KD method which only uses the sample pairs from the joint distribution, it can improve the accuracy better that our model also uses the sample pairs of the product of the marginal distributions to align the representation. With

Dataset	Method	Type	Architecture	Timestep	Accuracy
CIFAR-10	SpikeNorm (Sengupta et al. 2018)	ANN2SNN	VGG16	2500	91.55%
	Hybrid-Train (Rathi et al. 2020)	Hybrid training	VGG16	200	92.02%
	Spike-basedBP (Lee et al. 2020)	SNN training	ResNet11	100	90.95%
	STBP (Wu et al. 2019)	SNN training	CIFARNet	12	90.53%
	TSSL-BP (Zhang and Li 2021)	SNN training	CIFARNet	5	91.41%
	PLIF (Fang et al. 2021b)	SNN training	PLIFNet	8	93.50%
	GLIF (Yao et al. 2023)	SNN training	ResNet19	2	94.44%
	KDSNN (Xu et al. 2023)	SNN training	VGG16	4	91.05%
	Diet-SNN (Rathi and Roy 2020b)	SNN training	VGG16	5	92.70%
				10	93.44%
			ResNet20	5	91.78%
				10	92.54%
	STBP-tdBN (Zheng et al. 2020)	SNN training	ResNet19	2	92.34%
				4	92.92%
				6	93.16%
	TET (Deng et al. 2022)	SNN training	ResNet19	2	94.16%
				4	94.44%
				6	94.50%
	Our method	SNN training	VGG16	2	**93.83%**$\pm$0.10
				4	**94.27%**$\pm$0.09
			ResNet19	1	**95.33%**$\pm$0.09
				2	**96.08%**$\pm$0.10
			ResNet20	1	**92.16%**$\pm$0.07
				2	**93.40%**$\pm$0.08
				4	**94.27%**$\pm$0.07
CIFAR-100	BinarySNN (Lu and Sengupta 2020)	ANN2SNN	VGG15	62	63.20%
	Hybrid-Train (Rathi et al. 2020)	Hybrid training	VGG11	125	67.90%
	T2FSNN (Park et al. 2020)	ANN2SNN	VGG16	680	68.80%
	SNNThroughKD (Takuya et al. 2021)	SNN training	VGG16	5	74.42%
	Diet-SNN (Rathi and Roy 2020b)	SNN training	ResNet20	5	64.07%
			VGG16	5	69.67%
	TET (Deng et al. 2022)	SNN training	ResNet19	2	72.87%
				4	74.47%
				6	74.72%
	TEBN (Duan et al. 2022)	SNN training	ResNet19	2	75.86%
				4	76.13%
				6	76.41%
	GLIF (Yao et al. 2023)	SNN training	ResNet19	2	75.48%
				4	77.05%
	Our method	SNN training	VGG16	5	**76.37%**$\pm$0.11
			ResNet19	1	**77.75%**$\pm$0.08
				2	**78.79%**$\pm$0.10
			ResNet20	1	**67.93%**$\pm$0.12
				2	**69.81%**$\pm$0.12
				4	**72.86%**$\pm$0.10
CIFAR-DVS	Rollout (Kugele et al. 2020)	Rollout	DenseNet	10	66.80%
	STBP-tdBN (Zheng et al. 2020)	SNN training	ResNet19	10	67.80%
	RecDis-SNN (Guo et al. 2022b)	SNN training	ResNet19	10	72.42%
	LIAF-Net (Wu et al. 2022)	Conv3D	LIAF-Net	10	71.70%
	LIAF-Net (Wu et al. 2022)	LIAF	LIAF-Net	10	70.40%
	Real Spike (Guo et al. 2022c)	SNN training	ResNet19	10	72.85%
			ResNet20	10	78.00%
	Our method	SNN training	ResNet19	10	**80.00%**$\pm$0.20
			ResNet20	10	**78.50%**$\pm$0.10

Table 1: Comparison with SoTA methods on CIFAR.

Method	Type	Architecture	Timestep	Accuracy
Hybrid-Train (Rathi et al. 2020)	Hybrid training	ResNet34	250	61.48%
SpikeNorm (Sengupta et al. 2018)	ANN2SNN	ResNet34	2500	69.96%
STBP-tdBN (Zheng et al. 2020)	SNN training	ResNet34	6	63.72%
TET (Deng et al. 2022)	SNN training	ResNet34	6	64.79%
RecDis-SNN (Guo et al. 2022b)	SNN training	ResNet34	6	67.33%
OTTT (Xiao et al. 2022)	SNN training	ResNet34	6	63.10%
MS-ResNet (Hu et al. 2023)	SNN training	ResNet18	6	63.10%
Real Spike (Guo et al. 2022c)	SNN training	ResNet18	4	63.68%
		ResNet34	4	67.69%
SEW ResNet (Fang et al. 2021a)	SNN training	ResNet18	4	63.18%
		ResNet34	4	67.04%
Our method	SNN training	ResNet18	4	**62.95%**$\pm$0.08
		ResNet34	4	**66.78%**$\pm$0.10

Table 2: Comparison with SoTA methods on ImageNet.

Methods	Accuracy/Timestep=2	Accuracy/Timestep=4
baseline	68.27%	71.45%
Distillation	68.74%	71.72%
Contrastive loss(uniform weighted)	69.25%	72.16%
w/ SSCL	69.81%	72.86%

Table 3: Ablation experiments for the proposed contrastive loss and the CKA.

only 2 timesteps, our ResNet19 model also outperforms the TET (Deng et al. 2022) and the STBP-tdBN (Zheng et al. 2020) with 6 timesteps by 1.58%, and 2.92%, respectively. The same superiority can also be seen with ResNet20 backbone. These comparison results demonstrate the effectiveness and efficiency of our method.

CIFAR-100. We have also validated our approach on CIFAR-100. The results for CIFAR-100 are presented in Table 1. With 5 timesteps, our VGG16 model achieves an accuracy of 76.37%, which outperforms the SNNThroughKD (Takuya et al. 2021) by 1.95%. This once again demonstrates the superiority of our method over KD methods. Moreover, our method also achieves better accuracy than other previous works with fewer timesteps on ResNet19 and ResNet20. For instance, the accuracy of our ResNet19 model with just 1 timestep can be as high as 77.75%, while the GLIF (Yao et al. 2023) and the TEBN (Duan et al. 2022) are even less accurate at 4 time steps by 0.7% and 1.62%.

CIFAR10-DVS. The result for CIFAR10-DVS is shown in Table 1. Our method achieves 80.00% and 78.50% accuracy with ResNet19 and ResNet20 as the backbone respectively. It can be observed that the accuracy of our ResNet19 model is much higher than that of ResNet20 one. This can be explained from the perspective of mitigating overfitting. Since the data for training of CIFAR10-DVS is less sufficient than that of CIFAR10, and the overfitting issue is more severe on spiking ResNet19 model than ResNet20 one. Our contrastive learning method introduces the information of contrastive pairs from the product of marginal distributions and can be regarded as a proxy of data augmentation. Thus, it can alleviate the overfitting as well as improve the accuracy of spiking ResNet19 model more noticeably on CIFAR10-DVS.

ImageNet. All training settings for ImageNet are the same as CIFAR dataset but a temperature of 0.07 ($\tau = 0.07$) and training epoches as 320. We present the result for ImageNet in Table 2. It can be seen that our ResNet18 and ResNet34 model achieve 62.95% and 66.78% top-1 accuracy with only 4 timesteps, better than most of recent SoTA methods, only relatively smaller compared with SEW ResNet (Fang et al. 2021a) and Real Spike (Guo et al. 2022c). However, SEW ResNet (Fang et al. 2021a) and Real Spike (Guo et al. 2022c) are both SEW ResNet-based models, which are not typical SNN models. SEW ResNet-based models can fire positive integer spikes with the form of activation before addition. Since they have lost SNN's advantages of event-driven and multiplication-addition transform, we adopt the original spiking ResNet which fires standard binary spikes.

Conclusion

In order to enhance the representation of SNNs, this work proposes a new similarity-sensitive contrastive learning framework. Via Mutual Information (MI) maximization, the positive pairs of SNN's and ANN's representation of each layer from the same input samples will be pulled closer, while the negative pairs from different samples will be pushed apart. Futhermore, a similarity indicators (CKAs) for each layer is introduced to balance the layer-wise "push and pull" scheme. A series of ablation studies show that the proposed method can greatly increase the SNN's accuracy and will consistently outperforms the other SoTA methods.

References

Akopyan, F.; Sawada, J.; Cassidy, A.; Alvarez-Icaza, R.; Arthur, J.; Merolla, P.; Imam, N.; Nakamura, Y.; Datta, P.; Nam, G.-J.; Taba, B.; Beakes, M.; Brezzo, B.; Kuang, J. B.; Manohar, R.; Risk, W. P.; Jackson, B.; and Modha, D. S. 2015. 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*, 34(10): 1537–1557.

Bellec, G.; Salaj, D.; Subramoney, A.; Legenstein, R.; and Maass, W. 2018. Long short-term memory and learning-to-learn in networks of spiking neurons. *Advances in neural information processing systems*, 31.

Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K.; and Yuille, A. L. 2018. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4): 834–848.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Cortes, C.; Mohri, M.; and Rostamizadeh, A. 2012. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13(1): 795–828.

Davies, M.; Srinivasa, N.; Lin, T.-H.; Chinya, G.; Lines, A.; Wild, A.; Wang, H.; and Mathaikutty, D. 2018. Loihi: A Neuromorphic Manycore Processor with On-Chip Learning. *IEEE Micro*, 38: 82 – 99.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Deng, S.; Li, Y.; Zhang, S.; and Gu, S. 2022. Temporal Efficient Training of Spiking Neural Network via Gradient Reweighting. In *International Conference on Learning Representations*.

Duan, C.; Ding, J.; Chen, S.; Yu, Z.; and Huang, T. 2022. Temporal Effective Batch Normalization in Spiking Neural Networks. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Fang, W.; Yu, Z.; Chen, Y.; Huang, T.; Masquelier, T.; and Tian, Y. 2021a. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34: 21056–21069.

Fang, W.; Yu, Z.; Chen, Y.; Masquelier, T.; Huang, T.; and Tian, Y. 2021b. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, 2661–2671.

Feng, L.; Liu, Q.; Tang, H.; Ma, D.; and Pan, G. 2022. Multi-Level Firing with Spiking DS-ResNet: Enabling Better and Deeper Directly-Trained Spiking Neural Networks. *arXiv preprint arXiv:2210.06386*.

Girshick, R. 2015. Fast R-CNN. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 1440–1448.

Gretton, A.; Bousquet, O.; Smola, A.; and Schölkopf, B. 2005. Measuring statistical dependence with Hilbert-Schmidt norms. In *International conference on algorithmic learning theory*, 63–77. Springer.

Guo, Y.; Chen, Y.; Liu, X.; Peng, W.; Zhang, Y.; Huang, X.; and Ma, Z. 2023a. Ternary Spike: Learning Ternary Spikes for Spiking Neural Networks. *arXiv preprint arXiv:2312.06372*.

Guo, Y.; Chen, Y.; Zhang, L.; Liu, X.; Wang, Y.; Huang, X.; and Ma, Z. 2022a. IM-Loss: Information Maximization Loss for Spiking Neural Networks. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 156–166. Curran Associates, Inc.

Guo, Y.; Huang, X.; and Ma, Z. 2023. Direct learning-based deep spiking neural networks: a review. *Frontiers in Neuroscience*, 17: 1209795.

Guo, Y.; Liu, X.; Chen, Y.; Zhang, L.; Peng, W.; Zhang, Y.; Huang, X.; and Ma, Z. 2023b. RMP-Loss: Regularizing Membrane Potential Distribution for Spiking Neural Networks. *arXiv preprint arXiv:2308.06787*.

Guo, Y.; Peng, W.; Chen, Y.; Zhang, L.; Liu, X.; Huang, X.; and Ma, Z. 2023c. Joint A-SNN: Joint Training of Artificial and Spiking Neural Networks via Self-Distillation and Weight Factorization. *Pattern Recognition*, 109639.

Guo, Y.; Tong, X.; Chen, Y.; Zhang, L.; Liu, X.; Ma, Z.; and Huang, X. 2022b. RecDis-SNN: Rectifying Membrane Potential Distribution for Directly Training Spiking Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 326–335.

Guo, Y.; Zhang, L.; Chen, Y.; Tong, X.; Liu, X.; Wang, Y.; Huang, X.; and Ma, Z. 2022c. Real Spike: Learning Real-valued Spikes for Spiking Neural Networks. arXiv:2210.06686.

Guo, Y.; Zhang, Y.; Chen, Y.; Peng, W.; Liu, X.; Zhang, L.; Huang, X.; and Ma, Z. 2023d. Membrane Potential Batch Normalization for Spiking Neural Networks. *arXiv preprint arXiv:2308.08359*.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9729–9738.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hu, Y.; Deng, L.; Wu, Y.; Yao, M.; and Li, G. 2023. Advancing Spiking Neural Networks towards Deep Residual Learning. arXiv:2112.08954.

Hu, Y.; Wu, Y.; Deng, L.; and Li, G. 2021. Advancing residual learning towards powerful deep spiking neural networks. *arXiv preprint arXiv:2112.08954*.

Kornblith, S.; Norouzi, M.; Lee, H.; and Hinton, G. 2019. Similarity of neural network representations revisited. In *International conference on machine learning*, 3519–3529. PMLR.

Krizhevsky, A.; Nair, V.; and Hinton, G. 2010. Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5(4): 1.

Kugele, A.; Pfeil, T.; Pfeiffer, M.; and Chicca, E. 2020. Efficient Processing of Spatio-Temporal Data Streams With Spiking Neural Networks. *Frontiers in Neuroscience*, 14: 439.

Lee, C.; Sarwar, S.; Panda, P.; Srinivasan, G.; and Roy, K. 2020. Enabling Spike-Based Backpropagation for Training Deep Neural Network Architectures. *Frontiers in Neuroscience*, 14: 119.

Li, H. 2017. CIFAR10-DVS: An event-stream dataset for object classification. *Frontiers in Neuroscience*, 11.

Li, Y.; Kim, Y.; Park, H.; and Panda, P. 2023. Uncovering the Representation of Spiking Neural Networks Trained with Surrogate Gradient. *arXiv preprint arXiv:2304.13098*.

Lu, S.; and Sengupta, A. 2020. Exploring the Connection Between Binary and Spiking Neural Networks. *Frontiers in Neuroscience*, 14: 535.

Luo, X.; Qu, H.; Wang, Y.; Yi, Z.; Zhang, J.; and Zhang, M. 2022. Supervised learning in multilayer spiking neural networks with spike temporal error backpropagation. *IEEE Transactions on Neural Networks and Learning Systems*.

Nguyen, T.; Raghu, M.; and Kornblith, S. 2020. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. *arXiv preprint arXiv:2010.15327*.

Park, S.; Kim, S.; Na, B.; and Yoon, S. 2020. T2FSNN: Deep spiking neural networks with time-to-first-spike coding. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, 1–6. IEEE.

Rathi, N.; and Roy, K. 2020a. Diet-snn: Direct input encoding with leakage and threshold optimization in deep spiking neural networks. *arXiv preprint arXiv:2008.03658*.

Rathi, N.; and Roy, K. 2020b. DIET-SNN: Direct Input Encoding With Leakage and Threshold Optimization in Deep Spiking Neural Networks. *CoRR*, abs/2008.03658.

Rathi, N.; and Roy, K. 2021. DIET-SNN: A Low-Latency Spiking Neural Network With Direct Input Encoding and Leakage and Threshold Optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 1–9.

Rathi, N.; Srinivasan, G.; Panda, P.; and Roy, K. 2020. Enabling Deep Spiking Neural Networks with Hybrid Conversion and Spike Timing Dependent Backpropagation. In *International Conference on Learning Representations*.

Ren, D.; Ma, Z.; Chen, Y.; Peng, W.; Liu, X.; Zhang, Y.; and Guo, Y. 2023. Spiking PointNet: Spiking Neural Networks for Point Clouds. *arXiv preprint arXiv:2310.06232*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

Sengupta, A.; Ye, Y.; Wang, R.; Liu, C.; and Roy, K. 2018. Going Deeper in Spiking Neural Networks: VGG and Residual Architectures. *Frontiers in Neuroscience*, 13.

Simonyan, K.; and Zisserman, A. 2014. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv 1409.1556*.

Solomon, K. 1997. Information theory and statistics. *Courier Corporation*.

Song, L.; Smola, A.; Gretton, A.; Bedo, J.; and Borgwardt, K. 2012. Feature Selection via Dependence Maximization. *Journal of Machine Learning Research*, 13(5).

Takuya, S.; Zhang, R.; Nakashima, Y.; and . 2021. Training Low-Latency Spiking Neural Network through Knowledge Distillation. In *2021 IEEE Symposium in Low-Power and High-Speed Chips (COOL CHIPS)*, 1–3.

Tian, Y.; Krishnan, D.; and Isola, P. 2019. Contrastive representation distillation. *arXiv preprint arXiv:1910.10699*.

Tian, Y.; Krishnan, D.; and Isola, P. 2020. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, 776–794. Springer.

Wang, S.; Cheng, T. H.; and Lim, M.-H. 2022. LTMD: Learning Improvement of Spiking Neural Networks with Learnable Thresholding Neurons and Moderate Dropout. *Advances in Neural Information Processing Systems*, 35: 28350–28362.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; Xie, Y.; and Shi, L. 2019. Direct Training for Spiking Neural Networks: Faster, Larger, Better. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33: 1311–1318.

Wu, Z.; Xiong, Y.; Yu, S. X.; and Lin, D. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3733–3742.

Wu, Z.; Zhang, H.; Lin, Y.; Li, G.; Wang, M.; and Tang, Y. 2022. LIAF-Net: Leaky Integrate and Analog Fire Network for Lightweight and Efficient Spatiotemporal Information Processing. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11): 6249–6262.

Xiao, M.; Meng, Q.; Zhang, Z.; He, D.; and Lin, Z. 2022. Online Training Through Time for Spiking Neural Networks. arXiv:2210.04195.

Xu, Q.; Li, Y.; Shen, J.; Liu, J. K.; Tang, H.; and Pan, G. 2023. Constructing deep spiking neural networks from artificial neural networks with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7886–7895.

Yao, X.; Li, F.; Mo, Z.; and Cheng, J. 2023. GLIF: A Unified Gated Leaky Integrate-and-Fire Neuron for Spiking Neural Networks. arXiv:2210.13768.

Yin, B.; Corradi, F.; and Bohté, S. M. 2020. Effective and efficient computation with multiple-timescale spiking recurrent neural networks. In *International Conference on Neuromorphic Systems 2020*, 1–8.

Zhang, W.; and Li, P. 2021. Temporal Spike Sequence Learning via Backpropagation for Deep Spiking Neural Networks. arXiv:2002.10085.

Zheng, H.; Wu, Y.; Deng, L.; Hu, Y.; and Li, G. 2020. Going Deeper With Directly-Trained Larger Spiking Neural Networks. arXiv:2011.05280.