S^{4}NN: SCALABLE CONTRASTIVE SELF-SUPERVISED SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking Neural Networks (SNNs) offer a promising alternative to traditional artificial neural networks by leveraging sparse, event-driven computation that closely mimics biological neurons. When deployed on neuromorphic hardware, SNNs enable substantial energy savings due to their temporal and asynchronous processing. However, training SNNs remains fundamentally difficult because the non-differentiable nature of spike generation breaks the bidirectional gradient flow required in modern self-supervised learning (SSL) frameworks. In this work, we introduce the first scalable[1] fully SSL framework for SNNs that scales to large-scale visual tasks without requiring labeled fine-tuning. Our method leverages intrinsic spike-time dynamics by aligning representations across time steps and augmented views. To address gradient mismatch during surrogate training, we propose the MixedLIF neuron, which combines a spiking path with an antiderivative-based surrogate path during training to stabilize optimization, while retaining a fully spiking and energy-efficient architecture at inference. We also introduce two temporal objectives, Cross Temporal Loss and Boundary Temporal Loss, that align multi-time-step outputs to improve learning efficiency. Our approach achieves strong results across ResNet and Vision Transformer-based SNNs on CIFAR-10, CIFAR10-DVS, and ImageNet-1K. Our approach further generalizes through transfer learning from ImageNet-1K to downstream tasks, including image classification, as well as COCO object detection and instance segmentation. Notably, our self-supervised SNNs match or exceed the performance of some non-spiking SSL models, demonstrating both representational strength and energy efficiency.

1 INTRODUCTION

Spiking Neural Networks (SNNs) (Maass, 1997) are a class of biologically inspired models that process information through sparse, event-driven spikes (Pfeiffer et al., 2018), rather than continuous-valued activations. This asynchronous, spike-based computation allows SNNs to perform operations only when necessary, leading to significantly lower activity and reduced energy consumption. During inference, SNNs replace costly multiply-and-accumulate operations with simple accumulations, enabling orders-of-magnitude gains in efficiency. These benefits are particularly pronounced when deployed on neuromorphic hardware such as Intel's Loihi (Davies et al., 2021) and SynSense's Speck processors (Darshan et al., 2021), which are optimized for low-power, event-driven processing. However, training SNNs remains a significant challenge. The discrete nature of spike generation makes them non-differentiable and incompatible with standard backpropagation. Surrogate gradient methods (Neftci et al., 2019; Wu et al., 2018; Huh & Sejnowski, 2017; Bellec et al., 2018a) have enabled gradient-based training by approximating the spike function with smooth surrogates, leading to advances in supervised SNNs across convolutional and transformer architectures. However, these approaches remain reliant on labeled data and are yet to fully unlock the potential of SNNs in label-scarce settings.

In contrast, self-supervised learning (SSL) has transformed representation learning in ANNs by eliminating the reliance on manual labels and enabling models to extract generalizable features directly from raw data (He et al., 2022; Oquab et al., 2023; Caron et al., 2021; Zbontar et al., 2021). However, these advances have not transferred effectively to the spiking domain. Existing attempts

[1]Scalability denotes the ability of the SSL framework to train SNNs on large-scale datasets (e.g., ImageNet-1K) without diverging or requiring label supervision.

either depend on supervised fine-tuning after pretraining (Qiu et al., 2023; Hagenaars et al., 2021) or adapt ANN-style pretext tasks (Zhou et al., 2024b) without leveraging the distinctive temporal dynamics and sparsity of SNNs. As a result, prior SNN "SSL" methods remain limited to low-resolution benchmarks such as MNIST or CIFAR and do not generalize to high-resolution datasets or dense prediction tasks. Critically, no existing work has demonstrated a *fully self-supervised*, scalable SNN framework capable of operating at the level of ImageNet-1K or beyond.

This gap is particularly important because SSL addresses a fundamentally different scalability challenge than supervised training. Supervised SNNs rely on large annotated datasets, which are scarce for both neuromorphic sensors and real-world robotic environments. In contrast, SSL enables scalable representation learning directly from abundant unlabeled sensory streams—precisely the data regime where SNNs are most compelling due to their event-driven efficiency and temporal acuity. Solving SSL for SNNs therefore represents a critical step toward building label-efficient, scalable, and biologically grounded spiking models that can operate in realistic, continuously evolving environments.

Our Contributions. *We present the first fully self-supervised SNN framework that achieves competitive performance on large-scale pretraining (ImageNet-1K) and successfully transfers to large-scale downstream tasks such as COCO object detection and semantic segmentation, sometimes outperforming non-spiking SSL baselines*, without relying on labeled supervision. Our key insight is to exploit spike timing dynamics as a natural source of temporal diversity, enabling rich representation learning across time. We propose a dual-path contrastive learning framework that integrates a spiking path and an antiderivative-based surrogate path during training, aligning representations across time steps from two augmented views through temporal contrast. Only the spiking path is used at inference, preserving energy efficiency. We further propose two temporal alignment objectives that effectively learn from spike-time dynamics across augmented sequences: *Cross Temporal Loss*, which aligns all time steps, and *Boundary Temporal Loss*, which focuses on the first and final time steps to reduce computational cost. Our method is compatible with both CNN and Vision Transformer (ViT) based SNN architectures and demonstrates strong generalization on both static (ImageNet-1K, CIFAR-10) and neuromorphic (CIFAR10-DVS) datasets as well as strong transfer performance to downstream datasets. Notably, *we show that our self-supervised SNNs can outperform non-spiking SSL models in some settings, highlighting the representational advantage of SNN's temporal dynamics*, while also offering superior energy efficiency during inference.

2 Background & Related Work

2.1 Spiking Neural Networks

Spiking Neural Networks (SNNs) process information through discrete spike events over time, enabling sparse, event-driven computation that is attractive for energy-efficient learning systems and neuromorphic deployment (e.g., Loihi 2 (Davies et al., 2021)). In the context of this work, the key property of SNNs is their temporal state evolution—an attribute that offers potential advantages for representation learning across multiple augmented views, but also introduces the central technical challenge we address: the spike function is non-differentiable, preventing the bidirectional gradient flow required by modern self-supervised learning (SSL).

The LIF neuron is the standard computational unit for deep SNNs. Its discrete-time dynamics are

$$H_t = \tau V_{t-1} + W X_t, \quad S_t = \Theta(H_t - V_{\text{th}}), \quad V_t = (1 - S_t) \cdot H_t + S_t \cdot V_{\text{reset}}, \tag{1}$$

where H_t, V_t, and S_t denote the integrated current, membrane potential, and spike output at time t. These temporal updates are crucial in our SSL setting because they govern how information propagates across multiple time steps and across augmented views.

Despite the non-differentiability of $\Theta(\cdot)$, recent progress in surrogate-gradient (SG) learning has made supervised deep SNNs practical at scale (Neftci et al., 2019). SG methods replace the spike with a smooth surrogate during backpropagation, enabling stable optimization in CNN-based (Fang et al., 2021; Xiao et al., 2022; Meng et al., 2023; Du et al., 2025) and Transformer-based SNNs (Yao et al., 2023; Zhou et al., 2022b; Yao et al., 2024). However, all existing SG-trained SNNs operate strictly in *supervised* regimes, because the surrogate formulation still does not support the *bidirectional, cross-view gradient flow* required by contrastive or redundancy-reduction SSL objectives. This limitation directly motivates our MixedLIF design, which preserves surrogate-based differentiability while enabling reliable gradient exchange between augmented samples. Finally, on the deployment side, neuromorphic platforms such as Loihi 2, together with software stacks like `Lava-DL` (Team, 2023), provide efficient inference backends for SG-trained SNNs, further motivating scalable SSL frameworks that can exploit both event-driven computation and learned temporal structure.

2.2 Self-supervised Learning

SSL has become a dominant strategy in representation learning, especially in vision and language domains, due to its ability to learn from raw, unlabeled data. In the context of artificial neural networks (ANNs), SSL has matured into a powerful alternative to supervised training, achieving competitive performance on benchmarks such as ImageNet (Oquab et al., 2023). These models are often pretrained on large-scale unlabeled datasets and fine-tuned for downstream tasks, making them both scalable and versatile. However, top-performing methods still require billions of images, extensive data augmentations, and prolonged training times, limiting their practicality in edge and low-resource settings. SSL methods in ANNs have progressed from early contrastive objectives to more recent non-contrastive and reconstruction-based approaches. Early frameworks like SimCLR (Chen et al., 2020b) and MoCo (He et al., 2020; Li et al., 2021) used contrastive losses to align representations from augmented views while distinguishing them from other samples. Building on this, Hard Negative Mixing (HNM)(Kalantidis et al., 2020) enhanced contrastive learning by interpolating informative negative samples, and i-Mix(Lee et al., 2021) introduced a domain-agnostic label and feature mixing strategy to improve robustness. This line of work was later followed by non-contrastive methods such as BYOL (Grill et al., 2020b), and Barlow Twins (Zbontar et al., 2021), which eliminated the need for negative samples and focused instead on redundancy reduction or cross-view alignment. Parallel to these advances, masked image modeling has emerged as an effective pretext task. Inspired by BERT in NLP, methods like MAE (He et al., 2022), iBOT (Zhou et al., 2022a), MaskFeat (Wei et al., 2022), DINO (Caron et al., 2021; Oquab et al., 2023) and MSF (Koohpayegani et al., 2022) train models to reconstruct missing image patches or predict intermediate features from occluded inputs.

Despite these advances for ANNs, the application of SSL to Spiking Neural Networks (SNNs) remains limited and challenging (Zhou et al., 2024a). The fundamental impediment is the discontinuity in neuron responses during spike events, which prevents direct application of traditional SSL techniques to SNNs (Xu et al., 2021). While the biological plausibility of SNNs should theoretically make them ideal candidates for SSL (which better mimics how biological systems learn), their unique temporal dynamics create implementation barriers. Several recent works have attempted to bridge this gap. Qiu et al. (Qiu et al., 2023) introduced Temporal Contrastive Learning for SNNs, but their approach primarily focused on supervised learning with temporal constraints rather than pure self-supervision. Similarly, Hagenaars et al. (Hagenaars et al., 2021) applied SSL to event-based optical flow with SNNs, but still required supervised fine-tuning to achieve competitive results. The spiking SSL framework proposed in (Bahariasl & Kheradpisheh, 2024) achieves only 62% accuracy on CIFAR-10, and notably requires 20% of labeled data for fine-tuning. Singhal et al. (Singhal et al., 2024) proposed another contrastive SSL method for SNNs, but their approach is limited to neuromorphic datasets and relies on partial supervised fine-tuning of the entire network. Spikformer V2 (Zhou et al., 2024b) incorporated SSL for SNNs using masked image modeling but encountered instability issues with deeper networks. Most of these approaches attempt to directly transfer existing ANN-based SSL methods to SNNs without fully leveraging the inherent temporal characteristics of spike-based computation. Even with recent advances in supervised training methods for SNNs using SG (Eshraghian et al., 2023), the self-supervised domain lags behind.

3 Method

In this section, we describe our proposed SSL framework for SNNs. ~~As discussed above, training SNNs is inherently challenging due to the non-differentiability of spike generation and the associated gradient vanishing near the threshold.~~ While surrogate gradients mitigate spike discontinuity for supervised objectives, they do not preserve consistent cross-sample gradients required for self-supervised objectives. This motivates *MixedLIF*, a novel neuron module designed to stabilize surrogate-based training in self-supervised regimes. To further exploit the temporal structure of SNNs, we propose two loss functions: *Cross Temporal Loss* and *Boundary Temporal Loss*, which align representations across time steps.

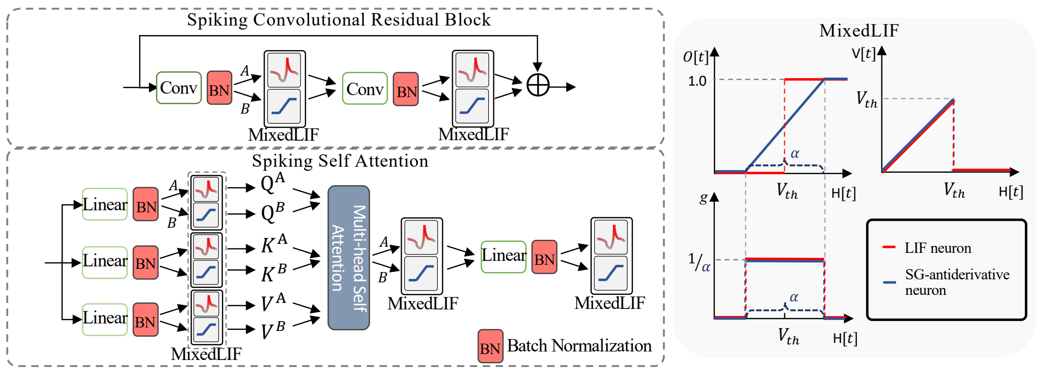

Figure 1: **Left:** Core components of our weight-shared dual-path SNN architecture. Both paths share the same trainable parameters but differ in activations via the MixedLIF neuron: Path A emits spikes using a standard LIF neuron, while Path B uses the SG-antiderivative neuron to update using true gradients. **Right:** Visualization of the MixedLIF neuron's dynamics, including input current $H_t[\cancel{t}]$, output spikes $O_t[\cancel{t}]$, post-spike membrane potential $V_t[\cancel{t}]$, and the surrogate gradient used for training.

3.1 MixedLIF

To develop an SSL framework for SNNs, we draw inspiration from the Barlow Twins (Zbontar et al., 2021) method. As illustrated in Figure 1, we present the *MixedLIF* neuron, a dual-path design where one path uses standard spiking dynamics and the other employs a range-continuous activation. This design enables gradient-based optimization and alleviates the gradient mismatch issue encountered in SNN training. *During inference, only the spiking path is used, preserving the energy efficiency inherent to SNNs.* In particular, our SSL framework comprises two paths, denoted as A and B, which process two independently distorted and time-augmented views, X^A and X^B, of the same input (see Figure 2). We adopt the same spatial augmentations as Barlow Twins (Zbontar et al., 2021) (random crop, flip, color jitter). Temporal augmentations such as time-reversal, frame dropout, and temporal jitter are applied only to event-based datasets where meaningful temporal structure exists. For static datasets, the input image is simply repeated across all timesteps, equivalent to direct encoding (Rathi et al., 2020), so temporal augmentations provide no benefit, as there is no inherent temporal information to exploit. Path A consists of standard LIF neurons. Path B, in contrast, uses the antiderivative of the surrogate function used in A, producing smooth, non-spiking activations to facilitate gradient flow. Specifically, the SG in path A is defined as

$$SG(H_t^A[\cancel{t}]) = \begin{cases} \frac{1}{\alpha} & , -\frac{\alpha}{2} \leq H_t^A[\cancel{t}] \leq \frac{\alpha}{2} \\ 0 & , \quad \text{otherwise} \end{cases} \tag{2}$$

where $H_t^A[\cancel{t}]$ is the accumulated input current at time step t, and α controls the width of the SG function, governing the steepness of the clipping-differentiable activation function. The antiderivative used in path B, computed by integrating ~~$SG(H[t])$~~ the SG function over the window $[V_{\text{th}}-\frac{\alpha}{2}, V_{\text{th}}+\frac{\alpha}{2}]$ is

$$ReLU_{\text{clip}}(H_t^B[\cancel{t}]) = \text{clip}\left(\frac{1}{\alpha}(H_t^B[\cancel{t}] - V_{\text{th}}) + \frac{1}{2}, 0, 1\right), \tag{3}$$

where $\text{clip}(x, 0, 1) = \min(\max(x, 0), 1)$ ensures that the output is bounded between 0 and 1. The outputs of the MixedLIF neuron for paths A and B are then given as

$$O_t^A[\cancel{t}] = \Theta(H_t^A[\cancel{t}] - V_{\text{th}}), \quad O_t^B[\cancel{t}] = ReLU_{\text{clip}}(H_t^B[\cancel{t}]). \tag{4}$$

The corresponding membrane potentials are then updated as

$$V_t^A[\cancel{t}] = (1 - O_t^A[\cancel{t}]) \cdot H_t^A[\cancel{t}] + V_{\text{reset}} O_t^A[\cancel{t}], \tag{5}$$

$$V_t^B[\cancel{t}] = \left(1 - \Theta\left(O_t^B[\cancel{t}] - \frac{1}{2}\right)\right) \cdot H_t^B[\cancel{t}] + V_{\text{reset}}\Theta\left(O_t^B[\cancel{t}] - \frac{1}{2}\right). \tag{6}$$

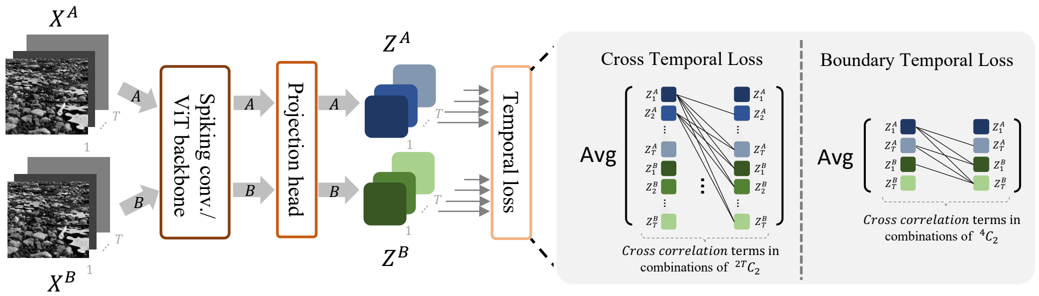
X^A
X^B
A
B
T
1
Spiking conv./ ViT backbone
Projection head
Z^A
Z^B
Temporal loss
Cross Temporal Loss
Avg
Cross correlation terms in combinations of $^{2T}C_2$
Boundary Temporal Loss
Cross correlation terms in combinations of $^{4}C_2$

Figure 2: Overview of our self-supervised training framework. Two independently distorted and time-augmented views, X^A and X^B, are passed through separate processing paths A and B. Path A uses Leaky Integrate-and-Fire (LIF) neurons to generate discrete spike outputs Z^A, while path B employs the antiderivative of the LIF surrogate function used in path A, yielding range-continuous outputs Z^B. Both sequences are then projected and compared via Cross Temporal or Boundary Temporal Loss to encourage temporally consistent and invariant representations.

Here we apply a hard reset for the spiking path and a refined hard reset-like mechanism for path B, setting V_{reset}=0 in both cases. We use hard resets rather than soft resets to avoid residual membrane activity, which could introduce distributional shifts between the two paths. Such shifts, if accumulated across layers, can degrade the effectiveness of the two-path SSL training.

Our framework is compatible with both CNN and ViT backbones (see Fig. 1), which we adapt into their spiking counterparts. We use 32-bit fixed point representation for the weights and membrane potentials. For the convolutional architecture, we follow the ResNet design, replacing ReLU with our MixedLIF neuron. For the ViT backbone, similar to Spikformer, we replace LayerNorm with BatchNorm, substitute GeLU with a spiking activation module (LIF in Spikformer, MixedLIF in our framework), and remove softmax attention. Notably, for Spikformer backbones, we obtain the final representation by averaging output features across the token dimension in the last stage. In contrast, convolutional backbones naturally produce a 1×1 spatial feature map due to progressive striding, eliminating the need for additional averaging. Additionally, we employ a non-spiking projection head following the Barlow Twins design (Zbontar et al., 2021) to enhance representation learning. This component is lightweight relative to the spiking backbone and adds negligible overhead ($<$1% in parame), preserving overall efficiency of the SNN. Converting the projection head to spiking incurs a 0.8% accuracy drop, providing a promising direction for achieving fully spiking architectures compatible with neuromorphic hardware.

3.2 Loss Functions

SNNs naturally encode information over time through asynchronous spike dynamics, providing an opportunity to leverage rich temporal representations that go beyond static frame-based learning. Traditional SSL methods such as Barlow Twins (Zbontar et al., 2021) operate on static embeddings by aligning representations from two augmented views of the same input. When naively extended to SNNs, this approach would involve aligning embeddings at corresponding time steps, meaning the first time step of path A is matched with the first time step of path B, and so on. However, such formulations overlook the temporal structure of spike-based computation, where information is not isolated per time step but distributed and causally linked across time. To fully exploit this temporally entangled structure, we propose the *Cross Temporal Loss*, a fundamentally spiking-aware objective that aligns representations across all pairs of time steps between the two augmented input streams. Rather than treating each time step as independent, our loss captures temporal cross-correlations that reflect how spike dynamics evolve and co-adapt over time (see Figure 2). This leads to feature representations that are not only robust to augmentations but also sensitive to the intrinsic causal and sequential structure of spikes, enabling the model to better generalize across time-varying inputs.

Formally, let Z_t^A and $Z_{t'}^B$ denote the output embeddings at time steps t and t' from two augmented views A and B, respectively. We compute the cross-correlation matrix $\mathcal{C}_{ij}(Z_t^A, Z_{t'}^B)$ between all pairs of time steps t and t' as

$$\mathcal{C}_{ij}(Z_t^A, Z_{t'}^B) = \frac{\sum_b z_{b,i}^{A,t}\, z_{b,j}^{B,t'}}{\sqrt{\sum_b (z_{b,i}^{A,t})^2}\,\sqrt{\sum_b (z_{b,j}^{B,t'})^2}} \tag{7}$$

where $z_{b,i}^{A,t}$ represents the scalar output embedding term in the channel dimension i of sample b in time step t in path A. The Cross Temporal Loss is then defined as

$$\mathcal{L}_{\mathcal{CT}} = \frac{1}{T} \sum_{\cancel{t=1}}^{\cancel{T}} \sum_{\cancel{t'=1}}^{\cancel{T}} \sum_{\{Z_t^p, Z_{t'}^{p'}\} \in \mathcal{P}(Z),\ Z_t^p \neq Z_{t'}^{p'}} \left[\sum_i \left(1 - \mathcal{C}_{ii}^2(Z_t^p, Z_{t'}^{p'})\right) + \lambda \sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2(Z_t^p, Z_{t'}^{p'}) \right] \tag{8}$$

where, T denotes the total number of time steps, and λ is a trade-off hyperparameter. The set $\mathcal{P}(Z)$ represents all multisets formed by pairing embeddings across time steps from both augmented views Z^A and Z^B, i.e.,

$$\mathcal{P}(Z) = \left\{ \{Z_t^p, Z_{t'}^{p'}\} \,\middle|\, p, p' \in \{A, B\},\ t, t' \in \{1, 2, \ldots, T\} \right\}.$$

This loss function promotes invariance by aligning the diagonal elements of the cross-correlation matrix to 1 and reduces redundancy by minimizing the off-diagonal elements.

However, the computational complexity of this loss scales quadratically with the number of time steps $O(T^2)$, which may increase the training complexity significantly for long sequences. To mitigate this, we also propose the *Boundary Temporal Loss*, which focuses on aligning representations at the initial (t=1) and final (t=T) time steps. As shown in Fig. 2, this provides a computationally efficient alternative while still capturing essential temporal dynamics. This boundary loss is formulated as

$$\mathcal{L}_{\mathcal{BT}} = \frac{1}{2} \sum_{t \in \{1,T\}} \sum_{t' \in \{1,T\}} \sum_{\{Z_t^p, Z_{t'}^{p'}\} \in \mathcal{P}(Z),\ Z_t^p \neq Z_{t'}^{p'}} \left[\sum_i \left(1 - \mathcal{C}_{ii}^2(Z_t^p, Z_{t'}^{p'})\right) + \lambda \sum_i \sum_{j \neq i} \mathcal{C}_{ij}^2(Z_t^p, Z_{t'}^{p'}) \right] \tag{9}$$

The proposed loss minimizes redundancy at the boundaries while retaining key temporal features. In Appendix A, we provide analytical support for why focusing solely on the boundary time steps may be sufficient to capture the dominant temporal structure in SNN representations.

Algorithm 1 Proposed Dual-Path SSL Training with MixedLIF

```
Require: shared parameters $\theta$, learning rate $\eta$, epochs $N$, batch size $B$, time steps $T$
 1: for epoch = 1 to $N$ do
 2:     for each batch $X$ of size $B$ do
 3:         $X^A, X^B \leftarrow$ spatial_augment($X$)
 4:         apply same temporal augmentation to $X^A, X^B$ over $T$ time steps (only for static datasets)
 5:         Initialize $\theta$ using Kaiming normal dist., Initialize membrane potential as $V_0^A[\cancel{0}]=0$ & $V_0^B[\cancel{0}]=0$
 6:         for $t = 1$ to $T$ do
 7:             $Z_t^A[\cancel{t}] \leftarrow$ proj_head(backbone($X_t^A[\cancel{t}], V_{t-1}^A[\cancel{t-1}]; \theta$))
 8:             $Z_t^B[\cancel{t}] \leftarrow$ proj_head(backbone($X_t^B[\cancel{t}], V_{t-1}^B[\cancel{t-1}]; \theta$))
 9:             Update $V_t^A[\cancel{t}]$ and $V_t^B[\cancel{t}]$ according to Eqs. 5 and 6
10:         end for
11:         $\mathcal{L}_{\mathrm{CT}} \leftarrow$ CrossTemporalLoss($Z^A, Z^B$) (using Eqs. 7 and 8)
12:         $\mathcal{L}_{\mathrm{BT}} \leftarrow$ BoundaryTemporalLoss($Z^A, Z^B$) (using Eqs. 7 and 9)
13:         Compute $g^A, g^B \leftarrow \nabla_\theta \mathcal{L}_{BT}$ (or $\nabla_\theta \mathcal{L}_{CT}$)
14:         $\theta \leftarrow \theta - \eta(g^A + g^B)$
15:     end for
16: end for
```

3.3 Training Mechanism

Our dual-path SSL framework, illustrated in Algorithm 1, improves the stability of SSL in SNNs by aggregating learning signals from both spiking activations in path A and surrogate-based activations in path B. Note that both paths share the same trainable parameters and process the same input

Table 1: Top-1 and top-5 accuracies under linear evaluation on static datasets (ImageNet-1K, CIFAR-10), and a neuromorphic dataset (CIFAR10-DVS).

Dataset	Method	Backbone	Time steps	Acc.(%)	
				Top-1	Top-5
ImageNet-1K	SimCLR Chen et al. (2020b)	ResNet50	-	69.3	89.0
	MoCo v2 Chen et al. (2020c)	ResNet50	-	71.1	-
	Barlow-Twins (Zbontar et al., 2021)	ResNet50	-	**73.2**	**91.0**
	Ours	Spiking-ResNet50	4	69.5	89.3
	Ours	Spikformer-16-512	4	**70.1**	**89.9**
CIFAR-10	Barlow-Twins[1] (Zbontar et al., 2021)	ResNet34	-	84.2	-
	Barlow-Twins[1] (Zbontar et al., 2021)	ViT-4-384	-	83.9	-
	Contrastive SSL Bahariasl & Kheradpisheh (2024)	Spiking CNN	30	62.2	-
	Ours	Spiking-VGG16	4	81.9	-
	Ours	Spiking-ResNet34	4	**85.6**	-
	Ours	Spikformer-4-384	4	**84.9**	-
CIFAR10-DVS	Contrastive SSL (Singhal et al., 2024) (Supervised)	Spiking-ResNet-18	10	**64.1**	-
	Ours	Spiking-ResNet34	10	63.9	-
	Ours	Spikformer-4-384	10	61.2	-

sample in parallel, with each path maintaining its own neuron dynamics. All weights are initialized identically using Kaiming normal distribution (He et al., 2015), ensuring symmetry at the start of training. Despite this, the difference in activation functions yields diverse gradient signals that enhance representation learning. The projected outputs from these paths are then used to compute the SSL objectives defined in Equations 8 and 9. During backpropagation, the gradients from both paths, g^A and g^B are aggregated to update the shared parameters: $\theta \leftarrow \theta - \eta(g^A + g^B)$, where η is the learning rate. This gradient fusion mechanism allows updates to be informed by both discrete spike dynamics and continuous surrogates, improving training stability and convergence. Our framework is agnostic to the specific surrogate function; any differentiable or piecewise-smooth approximation can be used. We also explored learning the surrogate slope α and firing threshold V_{th}, but found the gains to be marginal. Thus, we fix these values and focus on the architectural benefits of our dual-path, shared-weight design.

4 Results

To validate our approach, we pretrain our Spiking Neural Networks (SNNs) on ImageNet-1K (Deng et al., 2009), CIFAR-10 (Krizhevsky et al.), and the neuromorphic CIFAR10-DVS dataset (Li et al., 2017), using Spiking ResNet and Spikformer backbones. For clarity, we note “Spikformer-N-D” indicating the Spikformer is configured with patch size of N, and the feature embedding dimension is D. All models are pre-trained for 1000 epochs using momentum SGD (Ruder, 2016) with a learning rate (LR) of 0.005 and weight decay (WD) of $1.5e{-}6$ on Imagenet-1K, LR of 0.001 and WD of $1e{-}6$ on CIFAR-10 and CIFAR10-DVS. To prevent early-stage gradient explosion, we apply a linear warmup over the first 20 epochs and then decay the learning rate according to a cosine schedule. All experiments were conducted on 4 NVIDIA L40S GPUs, each with 48 GB of memory. Additional pre-training, fine-tuning, and transfer learning details are provided in Appendix B.

4.1 Linear Evaluation

We assess the quality of the learned representations by training a linear classifier on top of frozen features extracted from our dual-path spiking SSL model, trained with the Boundary Temporal Loss for its efficiency. This follows the standard linear evaluation protocol (Zhang et al., 2016; Oord et al., 2018; Bachman et al., 2019), as adopted in (Chen et al., 2020a). As shown in Table 1, our method achieves competitive performance on ImageNet, matching Barlow Twins on ResNet-50 with Spiking-ResNet50 (73.01% vs. 69.5%), and reaching 70.1% with Spikformer-16-512. On CIFAR10-DVS, our Spiking-ResNet34 and Spikformer-4-384 models achieve 63.9% and 61.2% accuracy, respectively. In comparison, (Singhal et al., 2024) reports 64.1% using a Spiking-ResNet18, but their method requires supervised fine-tuning of the entire network, whereas ours relies solely on linear evaluation. For CIFAR-10, Spiking-ResNet34 and Spikformer-4-384 attain 85.6% and 84.9%, outperforming the non-spiking Barlow Twins models (84.2% and 83.9%, respectively). These results demonstrate that our dual-path SSL framework produces high-quality spiking representations that rival, and in some cases surpass non-spiking self-supervised baselines.

[1]indicates self-implementation due to the unavailability of reported results on these datasets.

Table 2: Semi-supervised learning on ImageNet-1K and CIFAR-10 using 1% and 10% labelled training samples (Top-1 and Top-5 accuracy).

Dataset	Method	Backbone	Time steps	1% Top-1	1% Top-5	10% Top-1	10% Top-5
ImageNet-1K	SimCLR	ResNet50	-	48.3	75.5	65.6	87.8
	Barlow-Twins	ResNet50	-	**55.0**	**79.2**	**69.7**	**89.3**
	Ours	Spiking-ResNet50	4	52.6	77.5	67.2	88.3
	Ours	Spikformer-16-512	4	51.7	76.4	66.6	88.1
CIFAR-10	Barlow-Twins[1]	ResNet34	-	73.6	-	85.6	-
	Ours	Spiking-VGG16	4	73.9	-	85.9	-
	Ours	Spiking-ResNet34	4	**75.1**	-	**87.6**	-
	Ours	Spikformer-4-384	4	74.5	-	87.1	-

4.2 Semi-supervised Learning

Following Zhai et al. (2019), we randomly sample 1% and 10% of the labeled training images from each dataset and fine-tune the entire pretrained model, without any additional regularization, on these small subsets. Table 2 reports accuracies under these extremely low-label regimes. With just 1% labels on ImageNet, Spiking-ResNet50 is comparable to Barlow-Twins by (52.6% vs. 55.0% Top-1). At 10% labels, it also almost matches Barlow-Twins (67.2% vs. 69.7% Top-1). On CIFAR-10, our Spiking ResNet-34 model exceeds Barlow-Twins by over 1.5% (75.1% vs. 73.6%) with 1% labels and by 2% (87.6% vs. 85.6%) with 10% labels. These results confirm superior performance of our spiking SSL framework in label-scarce scenarios.

4.3 Transfer Learning

Image Classification: We evaluate the transferability of our self-supervised features on three standard benchmarks: CIFAR-10, CIFAR-100, and Oxford Flowers-102. For each dataset, we conduct both i) linear evaluation, training a linear classifier on frozen features, and ii) full fine-tuning of the entire network. As shown in Table 3, Spiking-ResNet50 achieves linear evaluation scores of 90.6% on CIFAR-10, 70.3% on CIFAR-100, and 90.2% on Flowers-102, closely matching the Barlow Twins baseline (91.1%, 71.6%, and 92.1%, respectively). With end-to-end fine-tuning, Spiking-ResNet50 reaches 96.4% on CIFAR-10, 81.2% on CIFAR-100, and 95.5% on Flowers-102, which are also comparable to Barlow Twins (97.9%, 85.9%, 97.5%). Spikformer-16-512 shows similar accuracy across both evaluation settings, but outperform Spiking-ResNet50 at flowers task.

Table 3: Transfer learning performance of the learned representations by our self-supervised learning pretrained on ImageNet-1K

	Method	CIFAR-10	CIFAR-100	Flowers
Linear evaluation	Barlow Twins-ResNet50[1]	**91.1**	**71.6**	**92.1**
	Ours-Spiking-ResNet50	90.6	70.3	90.2
	Ours-Spikformer-16-512	89.9	70.1	90.8
Fine-tune	Barlow Twins-ResNet50[1]	**97.9**	**85.9**	**97.5**
	Ours-Spiking-ResNet50	96.4	81.2	95.5
	Ours-Spikformer-16-512	96.3	82.3	96.1

Object Detection and Instance Segmentation: We evaluate performance by fine-tuning our ImageNet pretrained backbones on COCO (Lin et al., 2014) using the Mask R-CNN framework (He et al., 2017). Following the setup in (Chen et al., 2020b), all models adopt the C4 backbone variant (Wu et al., 2019) and are trained using the standard 1× schedule. As shown in Table 4, our Spikformer-16-512 achieves an AP of 37.8 for bounding-box detection (vs. 39.2) and 33.3 for instance segmentation (vs. 34.3), slightly underperform the Barlow Twins ResNet-34 baseline. The Spiking-ResNet50 model also performs competitively, further demonstrating the strong transferability of our spiking self-supervised representations to dense prediction tasks.

Table 4: Performance on COCO object detection and instance segmentation task by fine-tuning on ImageNet-1K pretrained backbone

	COCO det			COCO instance seg		
Method	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
Barlow Twins-ResNet50	**39.2**	**59.0**	**42.5**	**34.3**	**56.0**	**36.5**
Ours-Spiking-ResNet50	37.8	57.4	41.1	33.3	55.4	35.5
Ours-Spikformer-16-512	37.0	57.1	40.1	32.9	55.0	35.1

4.4 Ablation Studies

MixedLIF Training and Loss Functions: To assess the effectiveness of our MixedLIF module and proposed temporal losses, we conduct an ablation study on CIFAR-10 using linear evaluation. Specifically, we compare: (i) the full MixedLIF model with dual-path activations against a baseline vanilla LIF model using a single spiking path with hard reset, and (ii) three self-supervised loss variants—Cross Temporal Loss (CTL), Boundary Temporal Loss (BTL), and Non-Cross Temporal Loss (NCL), as summarized in Table 5. For Spiking-ResNet34 backbone, our MixedLIF with BTL achieves the highest accuracy of **85.6%**. Substituting BTL with CTL results in a minor decrease to 85.2%, while using NCL leads to a larger drop to 82.5%. Disabling MixedLIF and training with vanilla LIF and BTL reduces accuracy to 83.0%; adding CTL improves it slightly to 84.1%, while combining vanilla LIF with NCL yields 82.9%. Similar trends are observed for the transformer-based Spikformer backbone. MixedLIF with BTL achieves **84.3%**, with negligible degradation when using CTL (84.2%), and a more notable reduction with NCL (82.3%). In contrast, vanilla LIF with BTL achieves 82.1%, while pairing it with CTL improves accuracy to 82.9%. The combination of vanilla LIF and NCL produces the lowest result at 81.7%. These results collectively demonstrate that the SG-antiderivative neuron in MixedLIF is essential for stable and effective spiking self-supervised training, and that the Boundary Temporal Loss provides a favorable trade-off between computational efficiency and representation quality.

Table 5: Ablation study of loss and LIF-based activation by linear evaluation on CIFAR-10

Backbone	Method description	Acc. (%)
Spiking-ResNet34	MixedLIF + Boundary Temp Loss	**85.6**
	MixedLIF + Cross Temp Loss	85.2
	MixedLIF + Non-Cross Temp Loss	82.5
	LIF + Boundary Temp Loss	83.0
	LIF + Cross Temp Loss	84.1
	LIF + Non-Cross Temp Loss	82.9
Spikformer-4-384	MixedLIF + Boundary Temp Loss	**84.3**
	MixedLIF + Cross Temp Loss	84.2
	MixedLIF + Non-Cross Temp Loss	82.3
	LIF + Boundary Temp Loss	82.1
	LIF + Cross Temp Loss	82.9
	LIF + Non-Cross Temp Loss	81.7

Inference Latency: Figure 3 shows how varying the number of time steps T affects linear evaluation accuracy and inference latency for Spiking-ResNet34 and Spikformer-4-384, with $T{=}4$ on Spiking-ResNet34 as the latency baseline (1×). At $T{=}1$, both models perform poorly but run with low latency (0.28×/0.36×). Increasing to $T{=}2$ improves accuracy (28.5%/84.1%) at moderate cost (0.62×/0.81×). Accuracy saturates at $T{=}4$ (85.6%/84.9%), while $T{=}6$ yields minimal gains with substantially higher latency (1.57×/1.92×). Thus, $T{=}4$ achieves the best balance. Details of energy-efficiency and neuromorphic deployment are provided in Appendix D and C respectively.

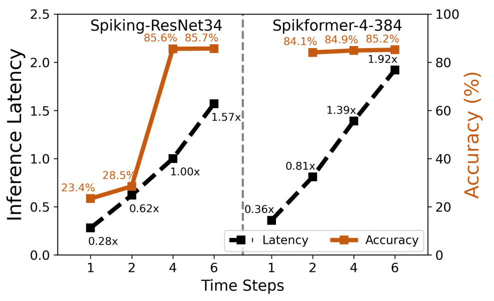

Figure 3: Top-1 accuracies (%) and inference latencies for different time steps under linear evaluation on CIFAR-10.

4.5 Training Efficiency

We evaluate the training latency of different loss functions on the CIFAR-10 dataset using two backbone architectures: Spiking-ResNet34 and Spikformer-4-384, as shown in Figure 4. For comparison, we introduce Non-Cross Temporal Loss (NCL), which computes cross-correlations only between matching time steps across the two augmented views (i.e., t of A with t of B). All latency values are reported relative to NCL on Spiking-ResNet34, which serves as the baseline (1×). For Spiking-ResNet34, the Cross Temporal Loss (CTL) incurs a significantly higher computational cost, increasing training time to 1.78×. In contrast, the Boundary Temporal Loss (BTL) results in only a slight increase (1.15×), indicating minimal overhead. With the Spikformer-4-384 backbone, NCL alone requires 1.41× more time than the

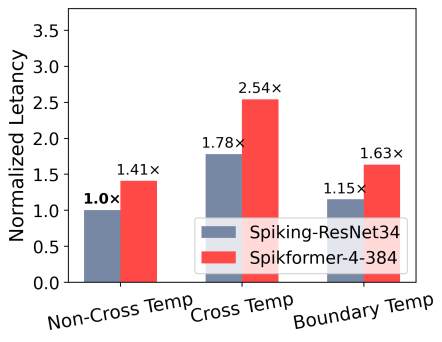

Figure 4: Training time comparison with Spiking-ResNet34 on CIFAR-10 between different loss functions. All values are normalized to the non-cross temporal loss.

Table 6: Comparison of supervised and SSL-trained SNNs under iso-architecture settings using $t = 4$ timesteps. CIFAR-10 reports Top-1 accuracy; ImageNet-1K reports both Top-1 and Top-5 accuracy.

Dataset	Architecture	Training Type	Top-1 (%)	Top-5 (%)
CIFAR-10	Spiking-ResNet34	Supervised	92.9	–
	Spiking-ResNet34	SSL	85.6	–
	Spikformer-4-384	Supervised	95.2	–
	Spikformer-4-384	SSL	84.9	–
ImageNet-1K	Spiking-ResNet50	Supervised	67.9	88.6
	Spiking-ResNet50	SSL	69.5	89.3
	Spikformer-16-512	Supervised	73.8	93.3
	Spikformer-16-512	SSL	70.1	89.9

Spiking-ResNet34 baseline. CTL raises this to $2.54\times$, whereas BTL reduces the overhead to $1.63\times$. This highlights the training efficiency of BTL, offering a favorable trade-off between speed and performance, as shown below.

4.6 Comparison with Supervised SNN Baselines

To contextualize the effectiveness of our contrastive self-supervised framework, we compare our SSL-pretrained SNNs against fully supervised SNNs trained under identical architectural configurations (iso-architecture) and identical temporal dynamics ($t = 4$ timesteps). Table 6 reports results for both CIFAR-10 (Spiking-VGG16 and Spikformer-4-384) and ImageNet-1K (Spiking-ResNet50 and Spikformer-16-512). On ImageNet, our SSL-trained SNNs achieve top-1 and top-5 accuracies within a small margin of their supervised counterparts; notably, SSL even *outperforms* supervised training by 1.6% for Spiking-ResNet50, while trailing by only 3.7% for Spikformer-16-512. These results demonstrate that label-free pretraining can learn high-quality, transferable representations at scale.

On smaller datasets such as CIFAR-10, SSL-trained SNNs initialized from scratch naturally underperform supervised baselines—a well-known trend mirrored in the ANN literature, where SSL methods typically require large-scale data to learn semantically rich features. However, when transferring our ImageNet-pretrained SSL SNNs to CIFAR-10, we observe substantial gains: the resulting fine-tuned models exceed the performance of most existing supervised-trained SNNs (see Table 3). This further underscores the value of SSL as a scalable, label-efficient pretraining strategy. Even when downstream datasets are small, the representations learned through large-scale, unlabeled SSL pretraining provide strong generalization benefits that supervised-from-scratch SNNs cannot match.

5 Conclusions

We present a fully self-supervised learning framework for Spiking Neural Networks that leverages mixedLIF neurons and temporal alignment to learn rich, temporally structured representations. Our method incorporates a dual-path architecture and two novel training objectives, i) Cross Temporal Loss, and ii) Boundary Temporal Loss, that are designed to exploit the sequential dynamics of spike-based computation. Through extensive experiments across standard vision and neuromorphic benchmarks, we demonstrated that our models not only match but in some cases outperform non-spiking SSL baselines such as Barlow Twins. This stands in contrast to prior SNN work, which typically narrows but does not close the performance gap with ANNs. We attribute this performance gain to the rich temporal signals captured by our architecture during training, which help optimize SNNs more effectively than frame-based methods. We believe our work opens a promising path toward scalable and energy-efficient self-supervised neuromorphic learning systems. Further advances in scalable training methods and hardware integration will be critical to enabling widespread deployment of self-supervised SNNs in real-world applications.

References

Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems*, 32, 2019.

Yeganeh Bahariasl and Saeed Reza Kheradpisheh. Self-supervised contrastive learning in spiking neural networks. In *2024 13th Iranian/3rd International Machine Vision and Image Processing Conference (MVIP)*, volume 1, pp. 1–5, 2024. doi: 10.1109/MVIP62238.2024.10491173.

Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *arXiv preprint arXiv:1803.09574*, 2018a.

Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert A. Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 795–805, 2018b. URL `https://papers.nips.cc/paper/7359-long-short-term-memory-and-learning-to-learn-in-networks-of//-spiking-neurons`.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PmLR, 2020b.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020c.

Manish Darshan, Shih-Chii Zhang, and Shuang Liu. Speck: A general-purpose neuromorphic processor with multi-core on-chip learning support. In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5. IEEE, 2021. doi: 10.1109/ISCAS51556.2021.9401391.

Mike Davies et al. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–934, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

S. Deng et al. Temporal efficient training of spiking neural network via gradient re-weighting. In *ICLR*, 2022. URL `https://openreview.net/forum?id=_XNtisL32jv`.

Kangrui Du, Yuhang Wu, Shikuang Deng, and Shi Gu. Temporal flexibility in spiking neural networks: Towards generalization across time steps and deployment friendliness. In *The Thirteenth International Conference on Learning Representations*, 2025. URL `https://openreview.net/forum?id=9HsfTgflT7`.

Jason K Eshraghian, Max Ward, Emre O Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023. doi: 10.1109/JPROC.2023.3308088.

Wei Fang et al. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. *CVPR*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *CoRR*, abs/2006.07733, 2020a. URL `https://arxiv.org/abs/2006.07733`.

Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020b.

Jesse Hagenaars, Federico Paredes-Vall'es, and Guido de Croon. Self-supervised learning of event-based optical flow with spiking neural networks. *Advances in Neural Information Processing Systems*, 34:10098–10110, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 10–14. IEEE, 2014.

Dongsung Huh and Terrence J. Sejnowski. Gradient descent for spiking neural networks. *arXiv preprint arXiv:1706.04698*, 2017.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21798–21809, 2020.

Soroush Abbasi Koohpayegani et al. Masked siamese networks for label-efficient learning. In *ECCV*, 2022.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). URL `http://www.cs.toronto.edu/~kriz/cifar.html`.

Kibok Lee, Yian Zhu, Kihyuk Sohn, Chun-Liang Li, Jinwoo Shin, and Honglak Lee. i-mix: A domain-agnostic strategy for contrastive representation learning. In *International Conference on Learning Representations*, 2021.

Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in neuroscience*, 11:309, 2017.

Siyuan Li, Zicheng Liu, Zedong Wang, Di Wu, Zihan Liu, and Stan Z Li. Boosting discriminative visual representation learning with scenario-agnostic mixup. *arXiv preprint arXiv:2111.15454*, 2021.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pp. 740–755. Springer, 2014.

Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. doi: 10.1016/S0893-6080(97)00011-7.

Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Towards memory- and time-efficient backpropagation for training spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6166–6176, 2023.

Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Maxime Oquab, Timoth'ee Darcet, Theo Moutakanni, Huy V Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

M. Pfeiffer et al. Deep learning with spiking neurons: Opportunities and challenges. *Frontiers in Neuroscience*, 12:774, 2018.

Haonan Qiu, Zeyin Song, Yanqi Chen, Munan Ning, Wei Fang, Tao Sun, Zhengyu Ma, Li Yuan, and Yonghong Tian. Temporal contrastive learning for spiking neural networks. *arXiv preprint arXiv:2305.13909*, 2023.

N. Rathi et al. DIET-SNN: Direct input encoding with leakage and threshold optimization in deep spiking neural networks. *arXiv preprint arXiv:2008.03658*, 2020.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

Raghav Singhal, Jan Finkbeiner, and Emre Neftci. Self-supervised pre-training of spiking neural networks by contrasting events and frames. In *UniReps: 2nd Edition of the Workshop on Unifying Representations in Neural Models*, 2024. URL https://openreview.net/forum?id=DNopfn4hZf.

Lava Team. Lava: A software framework for neuromorphic computing. 2023.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14668–14678, 2022.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12, 2018.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Lin Zhouchen. Online training through time for spiking neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22323–22335, 2022.

Xinyu Xu et al. Experimental demonstration of supervised learning in spiking neural networks with phase-change memory synapses. *Scientific Reports*, 11(1):1–13, 2021.

Man Yao, Jun Hu, Zhiyang Zhou, Liang Yuan, Yonghong Tian, Bing Xu, and Yi Yang. Spike-driven transformer. *Advances in Neural Information Processing Systems*, 36, 2023.

Man Yao, JiaKui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo XU, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=1SIBN5Xyw7.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arXiv:2103.03230*, 2021.

Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1476–1485, 2019.

Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pp. 649–666. Springer, 2016.

Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 11062–11070, 2021.

Changze Zhou, Hongze Zhang, Zhaofei Yu, Tiejun Ye, Lingjie Zhou, Tiejun Huang, Dongcheng Ma, Zhiwei Fan, Shiheng Zhou, and Yonghong Tian. Direct training high-performance deep spiking neural networks: a review of theories and methods. *Frontiers in Neuroscience*, 18:1383844, 2024a.

Daquan Zhou, Xiaohua Wei, et al. ibot: Image bert pre-training with online tokenizer. In *ICLR*, 2022a.

Z. Zhou et al. Spikformer v2: Join the high accuracy club on imagenet with an snn ticket. *arXiv preprint arXiv:2401.02020*, 2024b.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022b.

A Boundary Analysis

The *Boundary Temporal Loss* is motivated by the inherent temporal smoothness of spiking neural networks (SNNs), which arises from the leaky integration dynamics of neurons. For a standard Leaky Integrate-and-Fire (LIF) neuron, that does not spike with reset potential $V_{\text{reset}} = 0$, the membrane potential $H[t]$ evolves as:

$$H[t] = \tau H[t{-}1] + WX[t], \tag{10}$$

$$H[T] = \tau^{T-1} H[1] + \sum_{k=0}^{T-1} \tau^{T-k} WX[k] \tag{11}$$

This formulation shows that $H[t]$ behaves as a low-pass filtered version of historical input, evolving smoothly across time. As a result, the intermediate activations $H[2], H[3], \ldots, H[T{-}1]$ can be approximated as interpolations between the boundary states $H[1]$ and $H[T]$, with higher-order residuals. We further verified that adding intermediate timestep alignment provides marginal accuracy (<0.3%) but increases computation and memory access significantly, confirming the sufficiency of boundary sampling. Enforcing representation consistency at the start and end of the sequence thus implicitly regularizes intermediate steps due to the underlying dynamics. Moreover, these boundary states serve as critical temporal anchors. The initial state ($t = 1$) captures transient responses and high-frequency temporal features from the input, while the final state ($t = T$) encodes long-term dependencies through cumulative leaky integration. Together, they summarize the short- and long-term characteristics of spiking activity. Although this strategy does not explicitly supervise all time steps, its efficacy is supported by the sparse firing nature of SNNs. In our experiments, we measure an average spiking activity of approximately 23% across the network (see Figure. 5), indicating that neurons operate in subthreshold regimes most of the time. This ensures that hard resets, which could break temporal smoothness, occur infrequently. Consequently, the boundary representations retain most of the intermediate temporal information, enabling a biologically plausible and computationally efficient learning objective.

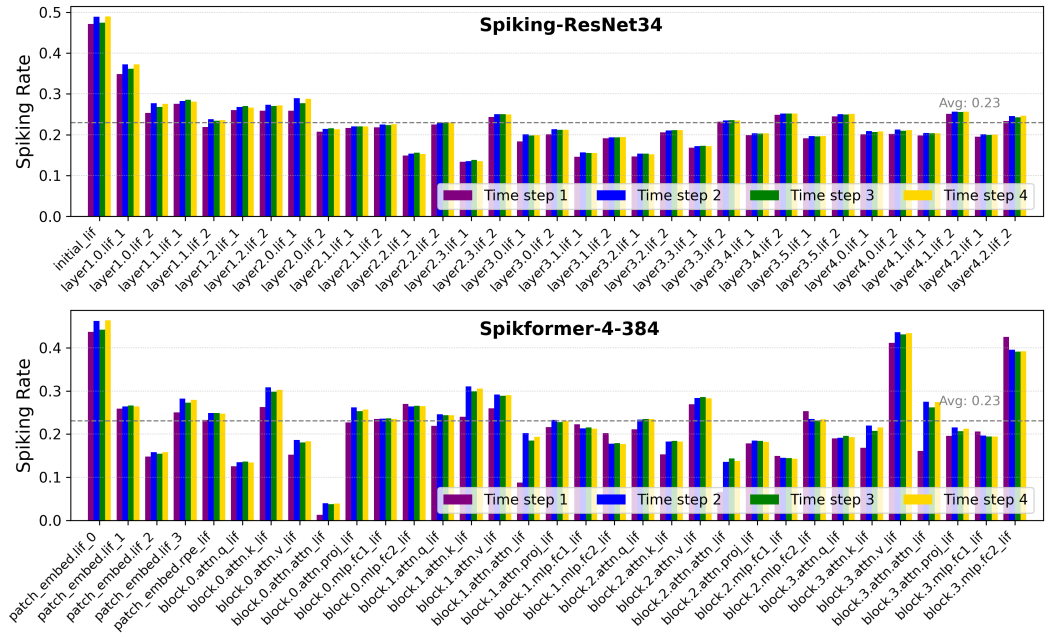

Figure 5: Layer-wise spiking rates over 4 time steps for Spiking-ResNet34 (top) and Spikformer-4-384 (bottom). Each cluster of four bars corresponds to one layer, with bar colors indicating time steps 1–4. The dashed horizontal line marks the average spiking rate across all layers and time steps.

B Experimental Setup

Data Preprocessing: In our experiments, we pretrain on ImageNet, CIFAR-10 and CIFAR10-DVS. For ImageNet and CIFAR-10 we apply the same spatial augmentations used in SimCLR (Chen et al., 2020b), BYOL (Grill et al., 2020a) and Barlow Twins (Zbontar et al., 2021): random crop and resize with horizontal flip, color distortion and Gaussian blur. Static images are then replicated into T time steps and resized to 224×224 for ImageNet. CIFAR10-DVS consists of 10000 event-based samples (converted from CIFAR-10) at 128×128. We apply temporal augmentation, including time-reversal, frame dropout, and temporal jitter for CIFAR10-DVS. Following prior work (Zheng et al., 2021), we split into 9000 training and 1000 test examples. Since these data already carry polarity and timestamp information, we omit color and temporal augmentations, load each sample with T=10 time steps, and resize frames to 48×48.

Default Pretraining Settings We use Spiking-ResNet50 and Spikformer-8-512 (patch size 8, embedding dimension 512) on ImageNet, each followed by a three-layer MLP projection head that maps to an 8192-dimensional space. For CIFAR-10 and CIFAR10-DVS we use a modified Spiking-ResNet34 in which the original 7×7, stride=2 convolution is replaced by a 3×3, stride=1 convolution and the following max-pooling layer is removed, and Spikformer-4-384 (patch size 4, embedding dimension 384) with a lighter two-layer projection head (2048-dim hidden layer followed by a 1024-dim output) for computational efficiency.

Linear Evaluation The linear evaluation follows the protocol of (Zhang et al., 2016), (Oord et al., 2018), and (Bachman et al., 2019). We freeze the pretrained backbone and train a linear classifier for 100 epochs, selecting hyperparameters for efficiency. We use AdamW, weight decay of 1e-6 and a cosine annealing schedule. The initial learning rate is set to 1e-4, and the batch size is 256. During training, each input is randomly cropped, resized to 224×224, and optionally horizontally flipped. At test time, images are resized to 256×256 then center-cropped to 224×224.

Semi-supervised Learning We train for 20 epochs using SGD with momentum (no weight decay). The base learning rate for both backbones is set to 1e-3, and 5e-2 for the final linear layer. We decay both rates by cosine annealing schedule. A batch size of 256 is used throughout.

Transfer Learning For linear classifier, we extract frozen features from the ImageNet pretrained network and train a linear classifier with SGD with learning rate of 1e-3 and weight decay of 1e-6. No data augmentation is applied. Input images are resized so the shorter side is 224, then center-cropped to 224 × 224. For fine-tuning, we initialize the full network with pretrained weights on ImageNet and fine-tune for 100 epochs with batch size 256, using SGD with momentum = 0.9. During fine-tuning, we apply only random resized crops and horizontal flips. At test time, images are resized with the shorter side 256 and then center-cropped to 224 × 224.

Object Detection The experiments are built on the Detectron2 framework (Wu et al., 2019). We initialize Mask R-CNN (He et al., 2017) with our ImageNet pretrained Spiking-ResNet34 and Spikformer-4-384 backbones. We train the C4 variant on the COCO 2017 training split and evaluate on the validation split. All hyperparameters follow Detectron2's standard 1× schedule, except that we set the base learning rate to 0.03.

C Neuromorphic Deployment

Our spiking models are designed to be directly deployable on neuromorphic hardware such as Intel's Loihi 2. During inference, the surrogate-based continuous path (Path B) used for training is discarded entirely. Only the event-driven spiking path (Path A), based on LIF neurons, is retained, making our approach compatible with neuromorphic execution and low-power deployment.

We implemented both the Spiking ResNet34 and Spikformer architectures in Lava-DL (Team, 2023), Intel's software framework for deploying SNNs on Loihi, and evaluated them on CIFAR-10, and CIFAR10-DVS. As shown in Table 7, the accuracy degradation from PyTorch simulation to Lava-DL execution is minimal (typically 0.4–1.2%), demonstrating the robustness of our models under deployment constraints. To ensure compatibility with Loihi's fixed-point hardware, we fold batch normalization layers into the preceding convolutional or linear layers during Lava-DL implementation. This standard practice removes the need for separate batch norm execution during inference while preserving its representational effect. Despite minor quantization and precision-related constraints,

Table 7: Accuracy (%) comparison between PyTorch simulation and Lava-DL deployment on Loihi.

Model	Dataset	PyTorch Sim.	Lava-DL (Loihi)
Spiking-ResNet34	CIFAR-10	85.6	84.9
Spiking-ResNet34	CIFAR10-DVS	67.3	66.1
Spikformer-4-384	CIFAR-10	84.9	84.2
Spikformer-4-384	CIFAR10-DVS	65.1	64.0

the performance remains competitive, highlighting the deployability of our self-supervised SNNs in real-world edge settings.

D TRAINING EFFICIENCY

To quantify the computational overhead introduced by the additional forward path in MixedLIF, we benchmark per-batch training cost under identical settings (Tiny-ImageNet, 224×224 input resolution, batch size = 128) using the same Barlow Twins framework. We report training time, energy consumption, and peak GPU memory for ResNet34, Spiking-ResNet34-LIF, and Spiking-ResNet34-MixedLIF. For fairness, both spiking variants are evaluated with a single timestep (t=1).

Table 8: Per-batch training cost comparison across network variants.

Method	Training Time (s)	Energy (J)	Peak Memory (MB)
ResNet34	**0.878**	**145.699**	**6219.26**
Spiking-ResNet34-LIF (t=1)	1.528	297.695	13702.16
Spiking-ResNet34-MixedLIF (t=1)	**1.465**	**281.468**	**10750.12**

Relative to the ANN baseline, Spiking-ResNet34-MixedLIF requires approximately 1.67× training time and 1.93× energy per batch. However, it remains consistently more efficient than the purely LIF-based model, reducing training time by 4.1%, energy consumption by 5.4%, and peak memory usage by 21.6%. These findings indicate that the two-path optimization strategy does not double the training cost; instead, the hybrid formulation leverages ANN pathways to reduce spike-driven computation and memory allocation, resulting in a substantially more favorable training profile compared to a full LIF architecture.

E INFERENCE EFFICIENCY

A key advantage of our proposed self-supervised SNN framework is its significantly lower energy consumption during inference, compared to dense ANN-based methods. While ANNs rely on multiply-and-accumulate (MAC) operations, SNNs compute using accumulate-only (AC) operations that are triggered by discrete spike events. This event-driven paradigm enables substantial reductions in energy consumption, particularly in sparsity-aware neuromorphic hardware, where inactive neurons and synapses incur no computational cost.

MAC operations in 32-bit fixed-point arithmetic consume approximately 3.1 pJ per operation in a 45nm CMOS process (Horowitz, 2014), whereas spike-driven accumulations typically require only 0.1 pJ, making them 31× more efficient. For example, as shown in Table 9, a ResNet-34 model trained using Barlow Twins on CIFAR-10 involves roughly 3.6 GFLOPs per inference, yielding an estimated compute energy of 11.16 mJ. In comparison, our Spiking-ResNet34 processes inputs over 4 time steps with an average 23% spike activity, leading to approximately 828 million active accumulations. This translates to an estimated compute energy of just 0.082 mJ, more than 136× lower than its ANN counterpart. Comparing our model to a purely LIF-based dual-path spiking baseline for SSL, we observe a 26% reduction (23% vs 29%) in spike activity on Spiking-ResNet34 and an 34% reduction (23% vs 31%) on Spikformer-4-384.

Beyond computation, memory access is a major contributor to energy consumption. ANNs must repeatedly load full activation maps and weights from memory, where each 32-bit SRAM access costs approximately 5 pJ, and DRAM access can exceed 100 pJ. In contrast, SNNs benefit from both activation sparsity and weight reuse. Since the same weights are applied across multiple time steps,

Table 9: Estimated inference-time compute for Barlow Twins and our spiking models on CIFAR-10. Energy is computed using 3.1 pJ/MAC for ANNs and 0.1 pJ/AC for SNNs based on (Horowitz, 2014). Spiking activity denotes the average proportion of active neurons over all the time steps.

Model	Type	Time Steps	Spiking Activity	Active Ops (M)	Energy (mJ)
Barlow Twins (ResNet34)	ANN-SSL	1	100%	3600	11.16
Spiking-ResNet34	MixedLIF SNN-SSL	4	23%	828	0.082
Spiking-ResNet34	LIF SNN-SSL	4	29%	1044	0.103
Spikformer-4-384	MixedLIF SNN	4	23%	1569	0.157
Spikformer-4-384	LIF SNN	4	31%	2115	0.212

they can be cached in local memory on neuromorphic or accelerator hardware, reducing redundant memory fetches. Additionally, spiking activations are sparse, so only a subset of neuron outputs are read or propagated at each step, further lowering bandwidth and memory energy usage on sparsity-aware systems. Moreover, since accumulation often occurs locally in neuromorphic implementations, write-back costs are reduced. As noted in Appendix C, during Lava-DL deployment, we fold batch normalization layers into the preceding convolution or linear layers, eliminating the need for runtime normalization without impacting performance.

Together, sparse compute, event-driven activations, and weight reuse, enable highly efficient inference. Our spiking models offer a compelling energy-performance trade-off, making them well-suited for real-time applications in edge and neuromorphic systems.

F Peak Memory Analysis

Table 10 compares the peak memory consumption of various models, their timestep configurations, and loss types. Notably, both Spikformer and ResNet34 models exhibit increased peak memory usage when trained with BTL and CTL loss functions, with CTL generally consuming the most. However, this increase remains within an acceptable range compared to non-cross-temporal training, which is the baseline for spiking SSL. Also, our MixedLIF neuron does not incur any additional peak memory overhead compared to the standard LIF neuron (LIF is used in both paths A and B), as they have similar activation dimensions for each layer in the network. Also, note that spiking models inherently consume more GPU memory than ANN models due to the need for temporal buffering and storage of multi-time-step representations.

Table 10: Absolute Peak Memory (MB) on CIFAR-10 (Batch size = 256) for Spiking-ResNet34 and Spikformer-4-384.

Method	Spiking-ResNet34	Spikformer-4-384
MixedLIF + NCTL	22,582.19	30,309.64
MixedLIF + BTL	23,044.84	32,310.34
MixedLIF + CTL	25,039.39	36,403.98
LIF + NCTL	22,854.77	30,738.41

Our experiments were conducted on CIFAR-10 with image resolution 32×32, using a batch size of 256 per GPU and a projection head dimension of 1024. For spiking models, we use 4 time steps, which naturally leads to approximately ∼4× higher peak memory usage for models trained with BTL or non-contrastive loss compared to the ANN baseline trained with Barlow Twins. Notably, our MixedLIF + BTL configuration exhibits peak memory usage comparable to the baseline LIF model with non-contrastive loss, demonstrating that each of our proposed components — MixedLIF neurons and the BTL loss — incur no more memory overhead than a typical SNN trained with Barlow Twins. The CTL-trained spiking models require additional memory due to the storage of all 4 4 cross-correlation terms during temporal contrastive loss computation. Peak memory was measured using PyTorch's `torch.cuda.max_memory_allocated()` immediately after each training step, and reflects memory used during forward, loss, and backward passes.

G Gradient Aggregation in MixedLIF

As shown in Figure 2 and detailed in Sections 3.2 of the paper, there is a single self-supervised loss computed using outputs from both Path A (spiking) and Path B (surrogate). While the two paths share identical network weights, they differ in their forward computations. Specifically,

1. Path A uses a threshold function during the forward pass, thereby forcing a surrogate gradient backpropagation with a clipped rectangular function.

Algorithm 2 Gradient Aggregation for MixedLIF

```
 1: for each batch $(X_A, X_B)$ in data loader do
 2:     $X_A, X_B \leftarrow$ augment(batch)                    ▷ Two augmentations of input
 3:     Forward pass for Path A (spiking)
 4:     $out_A \leftarrow$ model.forward_spiking($X_A$)
 5:     Forward pass for Path B (non-spiking, anti-derivative path)
 6:     $out_B \leftarrow$ model.forward_surrogate($X_B$)
 7:     Compute SSL loss with gradients only through Path A
 8:     $loss_A \leftarrow$ compute_ssl_loss($out_A, detach(out_B)$)
 9:     Backpropagate $loss_A$ and store gradients as $grad_A$
10:     Reset gradients
11:     Compute SSL loss with gradients only through Path B
12:     $loss_B \leftarrow$ compute_ssl_loss($detach(out_A), out_B$)
13:     Backpropagate $loss_B$ and store gradients as $grad_B$
14:     Aggregate gradients from both paths
15:     for each parameter $p$ in model do
16:         $p.grad \leftarrow grad_A + grad_B$
17:     end for
18:     Update model weights
19:     $optimizer.step()$
20:     Reset gradients
21: end for
```

2. Path B uses the antiderivative of this surrogate function (a clipped ReLU) during the forward pass, which results in smooth gradient signals.

Because of these differing forward computations, the gradients that each path produces during backpropagation are inherently different. These gradients are computed separately by detaching the complementary path in each case. This allows us to isolate clean gradient signals from both views. However, since both paths are processed in the same training pass and share parameters, their gradients are explicitly summed before the weight update step, rather than being treated separately or weighted. There is no weighting scheme involved. This design ensures that path A preserves the discrete spiking behavior critical for inference, and path B stabilizes training by providing dense gradient flow. The synergistic combination improves optimization, especially in the context of temporal self-supervised learning. We outline our algorithm for the gradient aggregation of the MixedLIF neuron below.

H CTL vs BTL Trade-off & Analysis

To better understand the representational trade-offs between the Cross Temporal Loss (CTL) and the Baseline Temporal Loss (BTL), we conducted a fine-grained representational analysis using two Spiking-ResNet34 models that one trained with CTL and the other with BTL. Both models were trained on CIFAR-10 with MixedLIF activation and an identical spiking backbone with total time steps of 4. Our goal was to go beyond runtime comparisons and assess whether CTL qualitatively encodes different temporal representations compared to BTL. To do this, we computed the per-timestep KL-divergence between the inference features generated by the two models over the entire CIFAR-10 test set. Specifically, for each timestep $t \in \{0, 1, 2, 3\}$, we extracted the latent feature vector z_t^{CTL} from the CTL-trained model and z_t^{BTL} from the BTL-trained model, and compared their distributions using histogram-based KL divergence:

$$D_{\mathrm{KL}}(\mathbf{p} \parallel \mathbf{q}) = \sum_{i=1}^{50} p_i \log\left(\frac{p_i}{q_i}\right) \tag{12}$$

Table 11: Per-Timestep KL Divergence

Timestep	Mean KL	Std. Dev.
1	0.251975	0.103575
2	0.259976	0.107579
3	0.257014	0.103883
4	0.259373	0.106984

where $\mathbf{p}$, $\mathbf{q}$ are the normalized histograms of activations from each model at a given timestep (50 bins, shared min/max range, and stabilized with small $\varepsilon = 1e-8$). The results are summarized

Table 12: Theoretical Training Time Complexity

Configuration	Forward	Backward	Loss	Explanation
Dual LIF + NCTL	2×	∼2×	∼T×	Sparse gradients in both paths. Efficient training.
MixedLIF + NCTL	∼2×	∼2×	∼T×	Path B has dense gradients from anti-derivative.
MixedLIF + BTL	2×	∼2×	∼6×	Loss only at t=0 and t=T. Constant cost w.r.t T.
MixedLIF + CTL	2×	∼2×	∼T(2T-1)×	Pairwise temporal loss over T×T steps.

in Table 11. The per-timestep KL divergences remain relatively stable across all time steps (with a narrow range between 0.251–0.260), indicating that CTL and BTL generate consistently similar global representations over time.

I MIXEDLIF VS LIF IN DUAL-PATH SETUP

In our framework, we explore two configurations for the dual-path encoder setup:

1. Both paths using LIF neurons (i.e., both use surrogate gradients based on clipped rectangular functions)
2. Path A using spiking LIF and Path B using a surrogate activation (MixedLIF) based on the antiderivative (clipped ReLU).

Both variants require two forward passes per batch (once through Path A and once through Path B), but their backward behavior differs. In dual-LIF, both paths produce sparse activations, enabling sparse gradient computation during backprop. In MixedLIF, Path B uses a non-sparse, differentiable surrogate path, making both the forward and backward pass denser and slightly more compute-heavy. However, the additional overhead of the time incurred during the forward and backward passes in MixedLIF may be negligible as evidenced by our empirical results shown below, because modern GPUs may not leverage irregular spike sparsity very well.

Additionally, the choice of temporal loss directly impacts both compute cost and memory complexity:

1. Non-Cross Temporal Loss (NCTL) computes independent contrastive losses at each time step, resulting in O(T) scaling, and T loss terms.
2. Boundary Temporal Loss (BTL) computes loss only between the first and last time steps, avoiding iteration over all time steps and keeping complexity constant with respect to T. It incurs $\binom{4}{2} = 6$ cross-correlation terms to calculate the loss value.
3. Cross Temporal Loss (CTL) computes pairwise contrastive loss across all (t_1, t_2) pairs, leading to O(T^2) computational complexity and the largest training cost. It incurs $\binom{2T}{2} = T(2T-1)$ cross-correlation terms to calculate the loss value, which is 28 for T=4.

These differences are further amplified in the dual-path setup, where loss is computed separately for each path and gradients are summed.

Our theoretical complexity estimates align closely with empirical results. Notably, forward pass durations remain consistent across configurations, suggesting that the structural differences (e.g., MixedLIF vs. LIF) do not significantly impact the forward path latency. In contrast, loss computation times differ substantially, with the CTL incurring significantly higher overhead (∼4.9×) compared to boundary or non-cross temporal loss as shown above. For simpler datasets, such as CIFAR10, a

Table 13: Per-Batch Training Time on ImageNet (Batch Size = 128)

Method	Forward	Backward	Loss	Total
Dual LIF + NCTL	0.382s	1.024s	0.003s	1.409s
MixedLIF + NCTL	0.381s	1.024s	0.003s	1.408s
MixedLIF + BTL	0.378s	1.089s	0.165s	1.632s
MixedLIF + CTL	0.376s	1.349s	0.814s	2.539s

similar trend can be observed as shown in Fig. 3 in the paper (CTL incurs 2.3-2.4x higher training time compared to BTL). This confirms that loss formulation, rather than neuron model choice, is the dominant factor in training efficiency. Therefore, practitioners seeking efficiency–accuracy trade-offs may prefer Boundary Temporal Loss as a middle ground, balancing temporal structure with low training complexity.

J Representation Learning in our Dual-Path Design

To investigate whether the two paths in the MixedLIF neuron—Path A (spiking) and Path B (continuous)—learn different internal representations despite sharing weights, we conducted an empirical analysis using two standard similarity metrics: cosine similarity and KL divergence. We forward the same input (with identical augmentations) through both Path A and Path B. To avoid mutual influence during gradient computation, we detach one of the paths and only allow backpropagation through the other. This setup ensures a clean and fair measurement of representational difference without interference from simultaneous weight updates. Let g_A and g_B denote the intermediate gradients tensors obtained from Path A and Path B, respectively. Cosine similarity is computed after flattening the tensors over the spatial, channel, and batch dimensions as shown below:

$$\text{cos\_sim}(g_A, g_B) = \frac{g_A^\top g_B}{\||g_A\||_2 \, \||g_B\||_2}$$

To measure distributional differences, we also computed the KL divergence between the normalized gradient histograms of g_A and g_B. Let $\mathbf{p}, \mathbf{q} \in \mathbb{R}^{50}$ denote the 50-bin histograms of g_A and g_B, normalized with a small $\varepsilon = 1e-8$ added for numerical stability:

$$\mathbf{p} = \frac{\text{hist}(g_A)}{\sum_{i=1}^{B} \text{hist}(g_A)_i} + \varepsilon, \quad \mathbf{q} = \frac{\text{hist}(g_B)}{\sum_{i=1}^{B} \text{hist}(g_B)_i} + \varepsilon$$

The KL divergence between $\mathbf{p}$ and $\mathbf{q}$ can be computed using Eq. 12. On average, cosine similarity is around 0.45 across intermediate layers, indicating moderate alignment in direction between Path A and Path B representations. Despite the moderate cosine similarity (0.45), this directional alignment suggests that both paths may optimize toward similar functional goals in parameter space. The difference in the learning space between path A and path B may not imply conflict but rather complementary diversity. Notably, the moderate KL divergence of 0.87 further confirms that Path B provides a meaningful correction signal—its continuous gradient helps guide Path A's spiking behavior, while staying semantically aligned. Path B's continuous gradients serve as a corrective signal, smoothing out the noisy or sparse updates from the spike-driven Path A, and helping guide shared weights more stably.

K Comparison with Other LIF Variants

We compare our proposed **MixedLIF** neuron with other popular LIF-based neuron models: i) **LIF**: Standard Leaky Integrate-and-Fire neuron, ii) **PLIF** (Deng et al., 2022): Trainable leak, iii) **ALIF** (Bellec et al., 2018b): LIF neuron with an adaptive threshold that increases after each spike and decays over time, and iv) **IF**: Integrate-and-Fire neuron with no leak term ($\tau = 0$). These models are evaluated under the same dual-path SSL setup with BTL. Unlike MixedLIF, these alternative neurons use the same neuron type in both paths. Thus, they cannot benefit from the full-precision surrogate gradient path provided by MixedLIF, which stabilizes training and provides temporally consistent gradient signals across paths. As a result, their accuracy drops significantly, with declines exceeding **2%** compared to MixedLIF, demonstrating that MixedLIF's dual-path design is essential for scaling SNNs to large-scale self-supervised learning tasks.

Table 14: Comparison of MixedLIF with alternative neuron models on CIFAR-10 under the BTL loss. The backbone used is Spiking-ResNet34.

Neuron Model	**Linear Eval Top-1 (%)**
MixedLIF	85.6
LIF	83.0
PLIF	83.4
ALIF	82.9
IF	81.3

As shown in Table 14, replacing MixedLIF with any of these alternatives results in a noticeable accuracy drop, confirming that MixedLIF is critical for achieving high performance in SSL settings. This highlights the importance of our design, which leverages the temporal dynamics of the non-spiking path to improve representation learning while preserving spiking sparsity during inference.

L VISUALIZATION

To further understand the impact of different training strategies on learned representations, we visualize the feature distributions using t-SNE (Van der Maaten & Hinton, 2008) projections. Figure 6 presents the t-SNE visualizations of the representations learned by different configurations at each time step. The baseline model using MixedLIF and Boundary Temperature Loss (Figure 6a) shows well-separated clusters with clear boundaries between classes, indicating strong discriminative features across all time steps. This aligns with its superior classification performance (85.6%) on the CIFAR-10 dataset.

The model trained with vanilla LIF (Figure 6c, d, and e) exhibits less defined clustering, particularly in earlier time steps. This suggests that the MixedLIF activation function helps establish more discriminative features in the temporal processing pipeline. The configuration using Non-Cross Temperature Loss model (Figure 6b and e) shows the least separation between clusters, consistent with its comparatively lower performance (82.9%). The visualization reveals that loss formulation contribute significantly to the quality of learned representations.

These visualizations collectively demonstrate how different combinations of activation functions and loss formulations influence the feature space organization across time steps, providing insights into why proposed MixedLIF and Boundary/Cross Temporal Loss achieve better classification performance than others.

M EFFECT OF RESET MECHANISMS ON TEMPORAL CONSISTENCY IN SSL

A key factor affecting the stability of self-supervised learning (SSL) in SNNs is the neuron reset mechanism. Standard *soft-reset* LIF neurons retain a residual membrane potential after emitting a spike, which introduces variability in the post-spike state. While this behavior is often beneficial for supervised tasks, we observe that in SSL it leads to inconsistent temporal statistics across augmented views. In particular, soft reset produces higher variance in spike-count distributions, causing fluctuations in the temporal dynamics that downstream contrastive objectives must align.

In contrast, the *hard-reset* mechanism used in MixedLIF directly resets the membrane potential to a fixed value after each spike, removing residual state differences between augmentations. This enforces more consistent temporal evolution across views and leads to significantly more stable spike-count distributions. As shown in Table 15, this stabilization improves the consistency of the SSL loss and yields a **+1.15%** accuracy improvement on CIFAR-10. Thus, by enforcing consistent post-spike states and reducing augmentation-induced variability, the hard-reset formulation in MixedLIF provides a more stable temporal signal that better supports cross-view learning.

Table 15: Impact of reset mechanism on temporal consistency and SSL performance for Spiking-VGG16 and Spikformer-4-384. Spike-count variance is computed across augmentations and normalized to the soft-reset baseline.

Model	**Reset Type**	**Spike Count Var.**	**SSL Loss Var.**	**Acc. (%)**
Spiking-VGG16	Soft Reset	1.00×	1.00×	81.1
Spiking-VGG16	Hard Reset	0.70×	0.78×	81.9
Spikformer-4-384	Soft Reset	1.00×	1.00×	83.4
Spikformer-4-384	Hard Reset	0.51×	0.63×	84.9

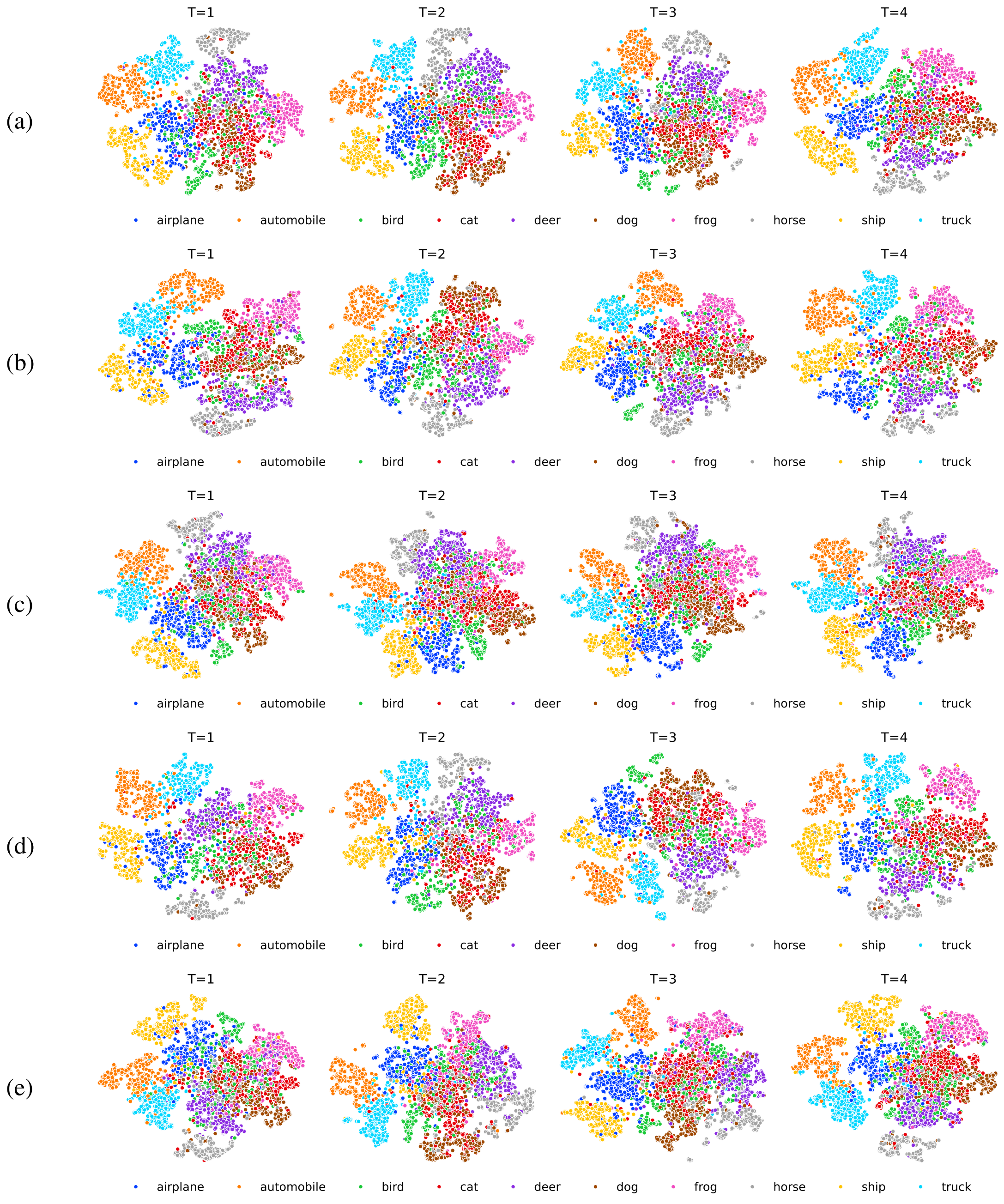

Figure 6: t-SNE visualization of Spiking-ResNet34 representations on CIFAR-10 for each ablation setting. (a) MixedLIF + Boundary Temporal Loss (baseline, 85.6% acc.), (b) MixedLIF + Non-Cross Temporal Loss (82.5% acc.), (c) LIF + Boundary Temporal Loss (83.0% acc.), (d) LIF + Cross Temporal Loss (84.1% acc.), and (e) LIF + Non-Cross Temporal Loss (82.9% acc.). Each row shows the progression of feature representation quality across time steps, with columns representing increasing time steps from left to right. Colors represent different CIFAR-10 classes. Note how higher-performing configurations display more distinct class clustering, especially in later time steps.