TEMPORAL-AWARE LEARNING: COORDINATING SUPERVISION WITH THE TEMPORAL EVOLUTION OF SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

The complex spatiotemporal dynamics of Spiking Neural Networks (SNNs) pose substantial challenges to direct training. In this work, we revisit this challenge from a temporal-evolution perspective by analyzing multi-granularity spiking activity, per-step predictions, and gradient relationships. We observe that spike firing fluctuations decrease, spatial firing patterns become more refined, and predictions become more accurate over time. Meanwhile, early-step gradients are less consistent with later-step gradients, whereas gradients among later steps become increasingly consistent. These observations suggest that temporal states contribute differently to optimization and uniform aggregation may underutilize coherent update tendency emerging at later steps. Unlike widely used objectives that supervise temporal states uniformly, we propose Temporal-Aware Learning (TAL), which considers the evolving states of SNNs. Our method incorporates quadratic temporal reweighting, later-step consistency regularization, and temporal target smoothing, to increase the contribution of locally coherent late-step gradients while softening early-step supervision. Experiments on static and neuromorphic datasets demonstrate that our method increases early–late gradient consistency, improves accuracy and accelerates convergence without additional inference overhead, with clear gains as temporal window increases within the evaluated range.

1 INTRODUCTION

Spiking Neural Networks (SNNs) have emerged as a brain-inspired computing paradigm that communicates information via discrete binary spikes (Maass, 1997; Wang & Yu, 2024). Their event-driven properties allow SNNs to perform sparse addition operations, obtaining reduced computational cost and energy consumption (Perez-Nieves et al., 2021; Yao et al., 2025; Ma et al., 2018). This paradigm has shown promise for neuromorphic sensing (Roy et al., 2019; Gallego et al., 2020), robotics (Adam D. Hines, 2025), and edge intelligence (Swaminathan, 2025) on platforms such as TrueNorth (Akopyan et al., 2015), Loihi (Davies et al., 2018), and Speck (Yao et al., 2024).

Directly training SNNs faces two main challenges: the non-differentiability of binary spike generation and complex spatiotemporal dynamics. The former has been largely addressed through spatiotemporal backpropagation based on surrogate gradients (Wu et al., 2018; Neftci et al., 2019; Shrestha & Orchard, 2018). Subsequent advances, including normalization (Kim & Panda, 2021; Zheng et al., 2021), residual connections (Fang et al., 2021a; Hu et al., 2024b), and attention mechanisms (Yao et al., 2021; 2023b), have further alleviated gradient vanishing in deeper SNNs. The latter remains a key obstacle, as preserving rich neuronal dynamics induces tight spatiotemporal coupling in both forward information flow and backward gradient propagation, entangling temporal states and making their gradient contributions difficult to coordinate during optimization.

Existing approaches to addressing this challenge posed by complex spatiotemporal dynamics are mainly as follows. First, reducing training complexity by decoupling gradient propagation across time steps, such as OTTT (Xiao et al., 2022) and gradient reversion (Hu et al., 2024a). Second, simplifying neuronal state updates at each time step by parameterizing decay, reset mechanisms, or membrane time constants (Fang et al., 2021b; Yao et al., 2022; Zheng et al., 2024). Third, preserving spatiotemporal computation while optimizing training process by per-step supervision (Deng et al., 2022), learnable membrane-potential initialization (Shen et al., 2024), surrogate-gradient adaptation (Liu et al., 2025), or temporal gradient calibration (Yan et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

However, existing supervision objectives treat temporal states equally at different time steps. In this work, we seek to understand and improve SNN training by investigating the evolution of temporal dynamics. Specifically, we jointly examine global firing-rate variation, spatial firing-rate maps, per-step predictions, and pairwise gradient relationships over time. The variance of global spike firing rates during early steps 1-3 is approximately $71\times$ that during later steps 7-10, indicating stronger fluctuations in early activity in Figure 1. The firing-rate maps gradually change from broadly distributed activation regions to more refined patterns, while per-step accuracy relatively increases over time in Figure 2 and 3. To understand optimization behavior underlying observations, we analyze pairwise gradient relationships across time steps in Figure 4. We reveal a structured evolution of gradient relationships: the average gradient cosine similarity between early and later steps is only -0.016 but reaches 0.74 among later steps, indicating opposing optimization directions across steps and increasing consistency at later steps. Such pairwise temporal gradient relationships across time steps have rarely been systematically characterized in existing SNN objectives. Since all temporal losses update same shared parameters, uniformly averaging gradients with different temporal relationships may underutilize the locally coherent optimization tendency at later steps.

To this end, we propose Temporal-Aware Learning (TAL), an optimization objective that coordinates supervision across evolving temporal states, consisting of three components. Quadratic temporal reweighting progressively increases later contributions to emphasize increasingly consistent gradient directions. Later-step consistency regularization is applied only to later-step outputs for regularizing local temporal variation, while temporal target smoothing relaxes early supervision and gradually restores one-hot targets at later steps. Extensive experiments validate the effectiveness of our method. For example, on CIFAR10DVS, it increases the mean pairwise gradient cosine similarity from 0.19 to 0.43, indicating more consistent update directions, while improving accuracy from 85.0% to 86.8%. Its accuracy increases from 84.2% at $T = 6$ to 87.4% at $T = 16$, with clearer relative benefits at the longer temporal windows evaluated. We make the following contributions:

- We characterize a time-dependent evolution where firing-rate variation decreases, spatial firing activity becomes more refined, and predictions improve over time, while pairwise gradients exhibit an underexplored temporal structure: early–late gradients show weak or negative consistency, whereas late–late gradients become more aligned. This suggests a mismatch between existing uniform supervision and the temporal evolution of SNNs.
- We propose Temporal-Aware Learning to match supervision with the evolving optimization characteristics of temporal states, integrating quadratic temporal reweighting to increase the contributions over time; later-step consistency regularization to promote consistent predictions among later steps; and temporal target smoothing to soften early targets.
- Extensive experiments validate the effectiveness of our method on both static and neuromorphic datasets without additional inference overhead. Further analyses demonstrate that our method improves early–late gradient consistency, accelerates convergence, and yields increasingly clear gains as the temporal window increases within the evaluated range.

2 Related Work

Direct training of SNNs. Existing SNN training approaches can be broadly categorized into unsupervised learning (Bi & Poo, 1998), ANN-to-SNN conversion (Diehl et al., 2015) and direct training. Among them, direct training typically employs backpropagation through time (BPTT) (Lee et al., 2016) with surrogate gradients to address the non-differentiability of binary spikes (Wu et al., 2018), and has achieved competitive performance (Luo et al., 2024). To improve gradient propagation and training stability in deeper SNNs, previous studies have introduced normalization techniques, such as BNTT (Kim & Panda, 2021) and tdBN (Zheng et al., 2021), as well as residual architectures, including SEW-ResNet (Fang et al., 2021a) and MS-ResNet (Hu et al., 2024b).

Enhancing temporal dynamics in SNNs. Existing studies have enhanced temporal representation in SNNs from neuronal dynamics and network architectures. At the neuronal level, parameterized spiking neurons adaptively learn membrane-related properties, enabling the modeling of diverse temporal patterns (Fang et al., 2021b; Mao et al., 2025). Various decay mechanisms have also been developed to capture complex spatio-temporal dependencies (Zhang et al., 2024). At the architectural level, temporal attention mechanisms dynamically adjust feature responses across time steps to enhance informative activations and suppress less relevant ones (Yu et al., 2022; Qiu et al., 2024).

Temporal optimization objectives in SNNs. Early direct training methods typically aggregate output membrane potentials across time and optimize prediction (Wu et al., 2018; Zheng et al., 2021). Subsequent studies introduced additional objectives, including membrane-potential distribution regularization (Guo et al., 2022b), entropy-based information maximization (Guo et al., 2022a), and temporal knowledge sharing (Dong, 2024). Recently, TET (Deng et al., 2022) further applies supervision to each time step, achieving better generalizability. Based on TET, recent studies have explored optimization from learnable initialization and distributions of membrane potential (Shen et al., 2024; Liu et al., 2025) and gradient calibration (Yan et al., 2025). However, existing works ignore pairwise gradient consistency evolution across time steps that motivates our work.

3 METHOD

In this section, we introduce two representative supervision paradigms for directly trained SNNs. We then investigate the temporal evolution of SNNs from multiple perspectives, and identify a potential mismatch between evolving temporal states and existing supervision schemes. Based on these, we propose Temporal-Aware Learning to coordinate supervision with the temporal evolution of SNNs.

3.1 EXISTING SUPERVISION FORMULATION

Directly trained SNNs commonly adopt two representative supervision paradigms. Sequence-level supervision (SLS) computes a classification objective from the outputs aggregated over the entire sequence with equal coefficients (Zheng et al., 2021; Zhang et al., 2025), whereas time-step-specific supervision (TSS) computes an objective for the prediction at each time step (Deng et al., 2022; Mukhoty et al., 2023), assigning equal coefficients to losses across time steps. For sequence-level supervision, temporal outputs are first aggregated to a sequence-level prediction:

$$L_{\mathrm{SLS}} = L_{CE}\left(\frac{1}{T}\sum_{t=1}^{T} O(t), y\right), \tag{1}$$

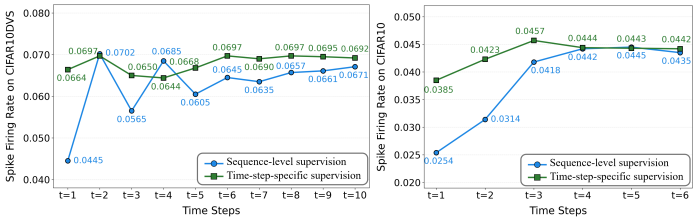
where $O(t)$ is pre-synaptic input of output layer at time step t, T is the total number of simulation steps, y is the ground-truth label, and L_{CE} is the cross-entropy loss. For time-step-specific supervision, the classification loss is computed separately at each time step and combined:

$$L_{\mathrm{TSS}} = \frac{1}{T}\sum_{t=1}^{T} L_{CE}\left(O(t), y\right). \tag{2}$$

3.2 MOTIVATION

To characterize the evolving temporal states, we examine existing supervision methods from firing-rate variation, feature maps, per-step predictions, and pairwise gradient relationships.

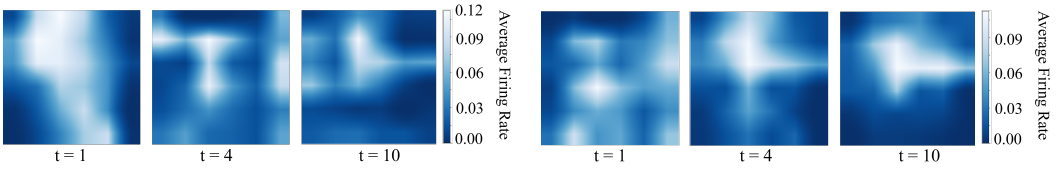
Observation 1: Global spike activity evolves from early fluctuations to later stability. As shown in Figure 1, global spike activity, computed by averaging spike firing rate over all neurons and layers at each time step, fluctuates substantially during early steps and gradually stabilizes at later steps. This pattern is consistently observed under both sequence-level and time-step-specific supervision methods on CIFAR10DVS and CIFAR10. For example, on CIFAR10DVS, the variance of spike firing rate over early steps 1–3 under sequence-level supervision is 71× that over later steps 7–10.



(a) Firing rate on CIFAR10DVS. (b) Firing rate on CIFAR10.

Figure 1: The evolution of spike activity in global network across time steps under sequence-level and time-step-specific supervision methods on CIFAR10DVS and CIFAR10.

Observation 2: Spatial spike activity becomes more concentrated in displayed examples. For each time step, we extract spike firing rate of penultimate layer and average them across all channels



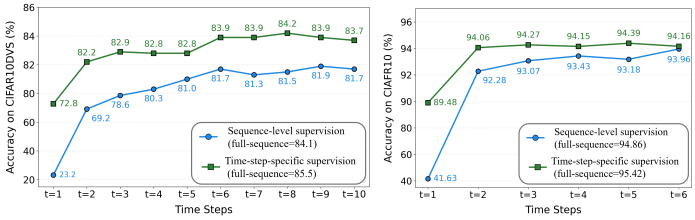
(a) Temporal evolution of spatial spike activity map under sequence-level supervision at different time steps on CIFAR10DVS.

(b) Temporal evolution of spatial spike activity map under time-step-specific supervision at different time steps on CIFAR10DVS.

Figure 2: Temporal evolution of penultimate-layer spatial spike activity under sequence-level and time-step-specific supervision methods on CIFAR10DVS using VGGSNN.

in a 2D spatial map. Each spatial location represents the proportion of channels that fire a spike at that location. As shown in Figure 2, early spike activity is broadly distributed across the feature maps, whereas later spike activity becomes concentrated in more specific regions.

Observation 3: Per-step prediction accuracy progressively improves. As shown in Figure 3, per-step classification accuracy is relatively low at early time steps, progressively improves over time, and eventually saturates at later steps. For example, on CIFAR10DVS, the accuracy gap between the first and last time steps reaches 58.5% under sequence-level supervision and 10.9% under time-step-specific supervision. Moreover, predictions aggregated over the full sequence consistently outperform those at any individual time step, indicating the benefit of temporal information integration.

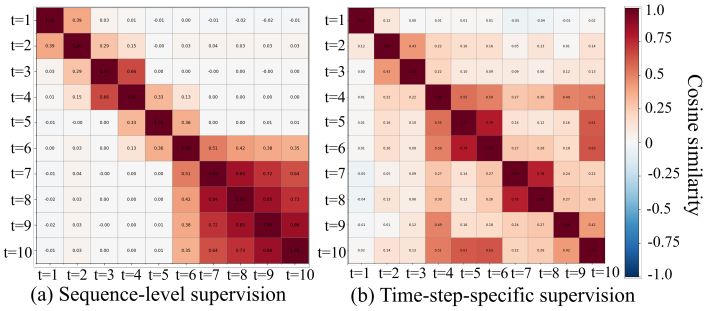


(a) Per-step and full-sequence prediction accuracy on CIFAR10DVS.

(b) Per-step and full-sequence accuracy on CIFAR10.

Figure 3: Temporal evolution of per-step accuracy and full-sequence prediction accuracy under sequence-level supervision and time-step-specific supervision methods.

Observation 4: Gradient relationships evolve from weak early–late to strong late-step consistency. We measure pairwise cosine similarities between diagnostic per-step gradients and treat steps 1-3 as early and steps 7-10 as later. Shown in Figure 4, sequence-level supervision yields average early-late and late-late similarities of -0.016 and 0.74, respectively, whereas the corresponding values under time-step-specific supervision are -0.01 and 0.34, respectively. The near-zero or negative early-late similarities indicate weak directional consistency, while the higher late–late similarities show stronger mutual directional consistency among later gradients, rather than establishing a globally optimal update direction. Under sequence-level supervision, diagnostic gradients reveal a similar variation, although they do not constitute an exact decomposition of the training gradient. Under time-step-specific supervision, uniformly weighted per-step losses may dilute locally coherent tendency at later steps.



(a) Sequence-level supervision

(b) Time-step-specific supervision

Figure 4: Pairwise cosine similarity between per-time-step gradients under sequence-level supervision and time-step-specific supervision on CIFAR10DVS at $T = 10$.

These observations indicate that temporal states exhibit different local optimization characteristics. This evolution is closely related to inherent spatiotemporal invariance of SNNs (Yao et al., 2023a), where spatial parameters are shared across time and same neuronal dynamics are repeatedly applied,

allowing similar background noise to persist at each step. This suggests a mismatch between existing supervision and the evolving dynamics of SNNs, motivating a training objective that progressively increases the relative contribution of later states with stronger local gradient consistency.

3.3 TEMPORAL-AWARE LEARNING

We propose Temporal-Aware Learning to coordinate supervision with temporal evolution, including quadratic temporal reweighting, late-step consistency regularization and temporal target smoothing.

3.3.1 QUADRATIC TEMPORAL REWEIGHTING

To differentiate the contributions of temporal states and reflect the stronger local gradient consistency observed at later time steps, we propose quadratic temporal reweighting (QTR), which assigns each time step t a normalized quadratic weight as:

$$w_t = \left(\frac{t-1}{T-1}\right)^2 + \delta, \quad \hat{w}_t = \frac{w_t}{\sum_{k=1}^{T} w_k}, \tag{3}$$

where $\delta > 0$ ensures a non-zero weight for the first time step. The reweighted objective is:

$$L_{QTR} = \sum_{t=1}^{T} \hat{w}_t L_{CE}\left(O(t), y\right), \tag{4}$$

$$\frac{\partial L_{QTR}}{\partial W} = \sum_{t=1}^{T} \hat{w}_t \cdot [S(O(t)) - \hat{y}] \cdot \frac{\partial O(t)}{\partial W}, \tag{5}$$

where L_{CE} is cross-entropy loss, $S(\cdot)$ is softmax function, and $\hat{y}$ is one-hot coding of ground-truth label y. Our hypothesis is neither that early-step gradients are uninformative nor that later-step gradients are globally optimal. Instead, we retain non-zero supervision for all states while moderately increasing the relative contribution of later states.

A. Rationale for increasing temporal reweighting. Temporal weighting strategy is motivated by the progressive integration of temporal information in SNNs. Ignoring thresholding and reset for illustration, subthreshold dynamics of a discrete-time LIF neuron with $u(0) = 0$ is unrolled as

$$u(t) = \sum_{k=1}^{t} \tau^{t-k} I(k), \tag{6}$$

where $I(k)$ denotes the input at time step k, and τ is the decay factor. Therefore, a later neuronal state depends on a longer input history rather than representing an independent response. Repeated temporal integration can further compound across computational stages. To illustrate this effect, we consider an N-stage cascade of unit-weight linear temporal integrators with a constant input I. Under $\tau = 1$, the state at the N-th stage can be recursively expressed as:

$$u^N(t) = I\binom{t+N-1}{N} \sim I\frac{t^N}{N!}, \qquad t \to \infty. \tag{7}$$

This simplified model does not determine the optimal weighting function for practical SNNs, which involve leakage, thresholding, resetting, nonlinear transformations, and time-varying inputs. It only indicates that later temporal states may integrate a longer computational history. We therefore adopt increasing temporal weighting as the general principle and instantiate it with a quadratic schedule, a simple convex design that distinguishes later states more clearly than linear weighting while remaining less concentrated than aggressively increasing schedules such as exponential weighting.

B. Gradient composition induced by temporal reweighting. Since quadratic temporal reweighting changes the relative contributions of per-step losses, we examine how this reweighting affects gradient composition. We next focus on time-step-specific supervision, for which the gradient composition can be written exactly as follows. Let $g_t(W) = \nabla_W L_t(W)$ denote training gradient induced by loss at time step t. At the same parameter point W, uniform per-step supervision and quadratic temporal reweighting combine same gradients using different coefficients:

$$g_{\mathrm{U}}(W) = \frac{1}{T}\sum_{t=1}^{T} g_t(W), \quad g_{\mathrm{QTR}}(W) = \sum_{t=1}^{T} \hat{w}_t g_t(W). \tag{8}$$

Uniform supervision assigns an equal coefficient to each temporal gradient, whereas quadratic temporal reweighting progressively shifts the relative contributions from earlier to later steps. To summarize the coherent tendency at later steps, we define late-step aggregated loss and gradient as:

$$L_{\text{late}}(W) = \frac{\sum_{j\in\mathcal{T}_{\text{late}}} L_j(W)}{|\mathcal{T}_{\text{late}}|}, g_{\text{late}}(W) = \frac{\sum_{j\in\mathcal{T}_{\text{late}}} g_j(W)}{|\mathcal{T}_{\text{late}}|}, \tag{9}$$

where $\mathcal{T}_{\text{late}}$ denotes later-step set in motivation analysis, and g_{late} serves as a local reference gradient that summarizes the aggregate tendency of later steps, rather than an assumed optimal direction. For a sufficiently small gradient update, stronger positive consistency with g_{late} corresponds to a larger first-order descent component for late-step aggregated loss. The difference between quadratic temporal reweighting and uniform weighting in the alignment with late-step aggregated direction is:

$$\langle g_{\text{late}}, g_{\text{QTR}}\rangle - \langle g_{\text{late}}, g_{\text{U}}\rangle = \sum_{t=1}^{T}\left(\hat{w}_t - \frac{1}{T}\right)\langle g_{\text{late}}, g_t\rangle . \tag{10}$$

Equation 10 characterizes how quadratic temporal reweighting changes the composition of aggregated gradient. It does not by itself establish that late-step reference direction is globally optimal. Instead, it shows that, under observed gradient structure, quadratic temporal reweighting increases the contribution aligned with locally coherent late-step direction. If $\langle g_{\text{late}}, g_t\rangle$ tends to increase with t, and $\left(\hat{w}_t - \frac{1}{T}\right)$ changes from negative at early steps to positive at later steps, then the weighted sum in Equation 10 is more likely to be positive. This provides a local and conditional interpretation of gradient composition rather than a claim of global optimality or a general convergence guarantee, since actual effect also depends on gradient magnitudes and optimization trajectory.

3.3.2 Later-step consistency regularization

To regularize local prediction variation among later temporal states, where gradients exhibit stronger mutual consistency, we introduce Later-step Consistency Regularization (LSCR):

$$L_{\text{LSCR}} = \frac{\sum_{t=t_{\text{later}}}^{T-1} ||O(t+1) - O(t)|\,|_2^2}{C(T - t_{\text{later}})}, \tag{11}$$

where C is the number of classes, and t_{later} is the starting point of later steps. This discourages abrupt oscillations after the prediction trajectory has entered a relatively stable stage, while leaving the rapidly evolving early states unconstrained by this consistency term. We regularize logits instead of probabilities since probability differences may become small under softmax saturation, whereas logits retain information about class margins.

3.3.3 Temporal target smoothing

To avoid imposing overly strict early-step supervision, we introduce Temporal Target Smoothing (TTS), which relaxes early-step targets while restoring the original one-hot supervision at later steps:

$$\hat{y}_t = \begin{cases} (1-\epsilon_t)\hat{y} + \epsilon_t \frac{\mathbf{1}}{C}, & 1 \le t \le t_{\text{early}}, \\ \hat{y}, & t > t_{\text{early}}, \end{cases} \tag{12}$$

where $\hat{y}$ is original one-hot label, t_{early} is the boundary of relatively early stage, $\mathbf{1} \in \mathbb{R}^C$ is an all-one vector, and C is the number of classes. The smoothing coefficient is linearly decayed as $\epsilon_t = \epsilon_{\max}\left(1 - \frac{t-1}{t_{\text{early}}-1}\right)$, where $\epsilon_{\max}$ controls maximum smoothing strength. Early targets receive smoothing that progressively diminishes over time, while one-hot targets are restored for $t \ge t_{\text{early}}$.

3.3.4 Overall optimization objective

Quadratic temporal reweighting determines the relative contribution of each temporal state, whereas temporal target smoothing determines the confidence of target. Combining these with later-step

consistency regularization, we obtain Temporal-aware learning, an optimization objective:

$$
\begin{aligned}
L_{TAL} &= \sum_{t=1}^{T} \hat{w}_t L_{CE}\left(O(t), \hat{y}_t\right) + \beta \cdot L_{LSCR}, \\
\frac{\partial L_{TAL}}{\partial W} &= \sum_{t=1}^{T} \hat{w}_t \cdot \left[S\left(O(t)\right) - \hat{y}_t\right] \cdot \frac{\partial O(t)}{\partial W} + \beta \cdot \frac{\partial L_{LSCR}}{\partial W},
\end{aligned}
\tag{13}
$$

where $\hat{y}_t$ is temporally smoothed target, and β controls the strength of consistency regularization.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETTINGS

All models are implemented in PyTorch and trained on NVIDIA Quadro RTX 8000 GPUs. Details of the datasets, network architectures, and training hyperparameters are provided in the **Appendix**. Since our method only modifies the training objective without introducing additional modules or trainable parameters, it incurs no computational or memory overhead during inference.

4.2 COMPARISONS WITH RECENT WORKS

Table 1: Comparisons with SOTA methods under different objectives on neuromorphic datasets.

Optimization Objectives	Method	Architecture	Time Step/Accuracy		
			CIFAR10DVS	DVSGesture	N-Caltech101
Sequence -Level Supervision	STBP-tdBN (Zheng et al., 2021)	ResNet19	10/67.8%	40/96.87%	-
	IMP (Shen et al., 2024)	VGG	16/86.3%	-	10/84.68%
	TCJA (Zhu et al., 2024)	VGGSNN	10/80.7%	20/99.0%	14/78.5%
	TKS-loss (Dong, 2024)	VGGSNN	10/85.3%	-	10/84.1%
	STAA (Zhang et al., 2025)	ResNet20	16/81.9%	-	-
		VGG13	16/82.10%	16/98.61%	-
	FSTA (Yu et al., 2025)	ResNet20	10/81.5%	-	-
			16/82.70%	-	-
Time-step -Specific Supervision	TET (Deng et al., 2022)	VGGSNN	10/83.2%	-	-
	TCJA-TET (Zhu et al., 2024)	VGGSNN	10/83.3%	20/98.2%	14/82.5%
	SLT (Anumasa et al., 2024)	VGGSNN	10/81.46%	10/98.43%	-
	LOCALZO (Mukhoty et al., 2023)	VGGSNN	10/81.87%	10/98.43%	10/82.99%
	DeepTAGE (Liu et al., 2025)	VGG11	10/81.23%	-	-
Temporal-aware learning	Ours	VGGSNN	**10/86.8%**	**10/98.96%**	**10/85.15%**
			16/87.4%	**20/99.31%**	**14/86.11%**

Table 2: Comparisons with SOTA methods on static datasets.

Method	T	Architecture	Accuracy		Architecture	Accuracy
			CIFAR10	CIFAR100		ImageNet-1K
MPD-loss (Guo et al., 2022b)	4	ResNet19	95.53%	74.10%	-	-
	6		95.55%	-	ResNet34	67.33%
IM-loss (Guo et al., 2022a)	6	ResNet19	95.49%	-	ResNet34	67.43%
	5	VGG16	93.85%	70.18%	VGG16	70.65%
TKS-loss (Dong, 2024)	4	ResNet19	96.35%	79.89%	SEW-ResNet34	69.60%
					SEW-ResNet50	70.70%
FSTA (Yu et al., 2025)	2	ResNet19	96.52%	80.42%	-	-
	4	ResNet20	94.72%	73.44%	ResNet34	70.23%
TET (Deng et al., 2022)	4	ResNet19	94.44%	74.47%	SEW-ResNet34	68.00%
	6		94.50%	74.72%	Spiking-ResNet34	64.79%
TMC (Yan et al., 2025)	6	ResNet19	-	78.05%	-	-
DeepTAGE (Liu et al., 2025)	4	ResNet18	95.86%	-	ResNet18	68.52%
Ours	4	MSResNet18	96.68%	80.97%	SEW-ResNet34	70.79%
	6		**96.90%**	**81.47%**		**71.08%**

As shown in Tables 1 and 2, our method achieves competitive performance on neuromorphic and static datasets, demonstrating the effectiveness of our method. Since prior works adopt different architectures and settings, these are intended to compare our method with published results, and fully controlled comparisons under identical architectures and training settings are in Sec.4.4. On

event-based datasets, it achieves 86.8%/87.4% on CIFAR10DVS at $T = 10/16$, 99.31% on DVS-Gesture, and 86.11% on N-Caltech101, comparing favorably with representative results from TKS-loss (Dong, 2024), TET (Deng et al., 2022), LOCALZO (Mukhoty et al., 2023), and DeepTAGE (Liu et al., 2025). On static datasets, our method obtains 96.68%/96.90% on CIFAR10, 80.97%/81.47% on CIFAR100, and 70.79%/71.08% on ImageNet-1K at $T = 4/6$, showing competitive performance relative to published results from recent TMC (Yan et al., 2025), and DeepTAGE (Liu et al., 2025). Although static images do not contain temporal variation, SNN states still evolve across time steps via membrane integration and leakage. The firing rate and per-step prediction on CIFAR10 in Figures 1(b) and 3(b) confirm that such temporal evolution also exists for static inputs.

4.3 Ablation and Analysis

Component-wise ablation. As shown in Table 3, we evaluate the contributions of three components. Each component improves performance on both datasets, with quadratic temporal reweighting providing the largest gains of 1.5% and 3.12% on CIFAR10DVS and N-Caltech101, respectively. Later-step consistency regularization improves accuracy by 0.4% and 0.69%, while temporal target smoothing improves accuracy by 0.8% and 1.17%. Combining all components achieves the best accuracies of 86.8% and 86.11%, showing that these components can be effectively combined.

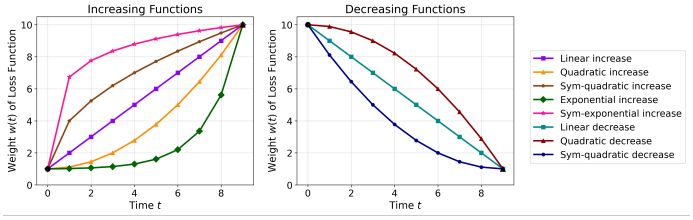
Table 3: Ablation study on our components.

QTR	LSCR	TTS	Accuracy	
			CIFAR10DVS	N-Caltech101
×	×	×	85.0%	82.81%
✓	×	×	86.5%	85.93%
×	✓	×	85.4%	83.50%
×	×	✓	85.8%	83.98%
✓	✓	✓	**86.8%**	**86.11%**

Temporal reweighting strategy. Figure 5 illustrates the contribution distributions of different weighting schedules, while Table 4 reports performance under identical training settings. Among the evaluated schedules, all increasing schedules outperform uniform weighting, whereas the evaluated decreasing schedules lead to lower accuracy, supporting the benefit of progressively increasing relative contribution of later states. Specifically, quadratic temporal reweighting achieves the highest accuracy, reaching 86.5% on CIFAR10DVS and 85.93% on N-Caltech101. It outperforms linear increasing weighting by 0.60% and 1.82%, respectively, suggesting that quadratic schedule provides a favorable empirical balance between maintaining early-step supervision and emphasizing later states.

Table 4: Accuracy of different temporal reweighting strategies on CIFAR10DVS and N-Caltech101.

Strategy	Accuracy	
	CIFAR10DVS	N-Caltech101
Uniform	85.0%	82.81%
Linear Increase	85.9%	84.11%
Quadratic increase (QTR)	**86.5%**	**85.93%**
Sym-quadratic increase	86.1%	85.15%
Exponential increase	86.0%	84.63%
Sym-exponential increase	85.9%	84.50%
Linear decrease	83.2%	81.51%
Quadratic decrease	84.7%	82.55%
Sym-quadratic decrease	84.5%	82.03%



(a) Increasing weighted function curves of loss function $w(t)$ along with time t ($T = 10$)

(b) Decreasing weighted function curves of loss function $w(t)$ along with time t ($T = 10$)

Figure 5: Temporal contribution distributions of different reweighting strategies.

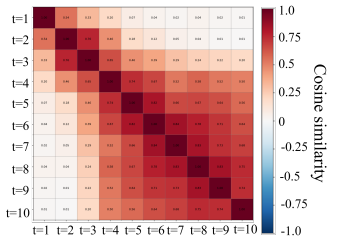


Figure 6: Pairwise gradient cosine similarity of our method.

Temporal Gradient Coordination Analysis. We examine gradient coordination by pairwise cosine similarity between gradients across time steps. As shown in Figure 6, our method yields higher pairwise gradient consistency than compared objectives. The average pairwise similarity increases from 0.19 and 0.22 to 0.43, while average early–late similarity rises from -0.016 and -0.01 to 0.08.

These results indicate that our method converges to parameter regions with stronger consistency. The improved alignment reflects the combined effect of the three components through the optimization trajectory, rather than a direct modification of gradient similarity at a fixed parameter point.

4.4 Model Analysis

To verify the effect of optimization objective, we compare our method with sequence-level and TET-style time-step-specific supervision under identical network architecture and training setting.

Effect of longer time window. On CIFAR10DVS shown in Figure 7, our method achieves 84.2%, 86.8%, and 87.4% at T= 6, 10, and 16. Although slightly below time-step-specific supervision at T=6, our method shows clearer gains at T=10 and T=16 within the evaluated temporal range, and surpasses sequence-level and time-step-specific supervision by 2.7% and 1.8% at T=10. A similar trend is observed on N-Caltech101, indicating that our method can benefit more when temporal evolution is more fully developed within the evaluated range.

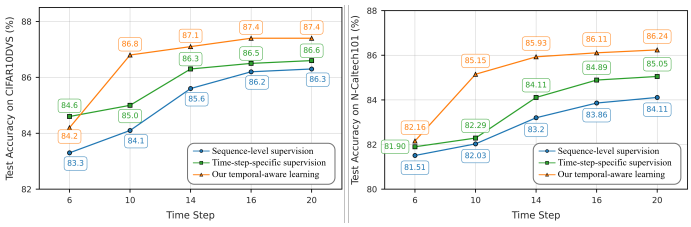


Figure 7: Test accuracy at different time windows using different supervision methods on CIFAR10DVS and N-Caltech101.

Convergence behavior. Figure 8 compares the training dynamics of different objectives, showing faster empirical convergence of our method under the evaluated settings. Our method reaches the target accuracy in fewer epochs, requiring 95 and 84 epochs on CIFAR10DVS and N-Caltech101, respectively, compared with 140/186 epochs for sequence-level supervision and 116/163 epochs for time-step-specific supervision.

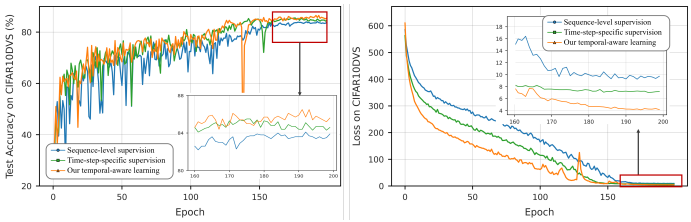


Figure 8: Test accuracy and loss curves over training epoch using different supervision methods on CIFAR10DVS.

Local loss geometry. We visualize loss landscapes around converged solutions (Li et al., 2018). Sequence-level, time-step-specific, and our method converge to local minima with losses of 0.0717, 0.0688, and 0.0219 on CIFAR10DVS, in Figure 9. Our method reaches a broader and flatter basin with a smoother neighborhood, indicating lower local sensitivity to the selected perturbation directions.

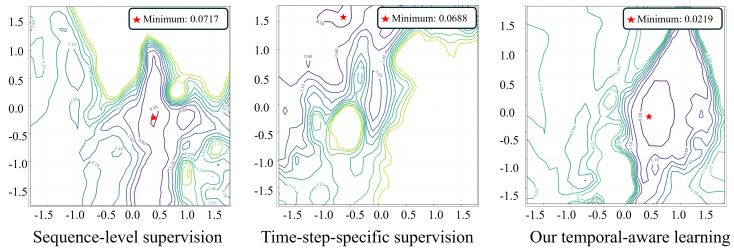


Figure 9: Loss landscapes around the converged solutions of different supervision methods on CIFAR10DVS.

5 Conclusion

In this paper, we reveal a mismatch between existing supervision and the temporal evolution of SNNs, and propose Temporal-Aware Learning to account for the evolving characteristics of temporal states. Our method integrates quadratic temporal reweighting, later-step consistency regularization, and temporal target smoothing to provide step-wise supervision and improve temporal gradient coordination. Extensive experiments on neuromorphic and static datasets demonstrate that our method improves performance and accelerates convergence. Moreover, it can be easily integrated into existing SNN architectures without additional inference overhead.

AI USE STATEMENT

We used generative AI tools to assist with English writing, translation from internal drafts, language polishing, and LaTeX/table formatting. We did not use generative AI tools to formulate the research idea, design the methodology or experiments, implement the proposed method, generate experimental results, or interpret the results. All AI-assisted content was reviewed and edited by the authors, who take full responsibility for the final submission.

ETHICS STATEMENT

This work proposes a temporal-aware training objective for spiking neural networks and does not involve human subjects, personal data collection, or the release of new datasets. All experiments are conducted on publicly available benchmarks. Potential risks are limited to those generally associated with advances in machine learning, and we encourage responsible use consistent with standard safety and privacy practices.

REPRODUCIBILITY STATEMENT

We provide the model formulation and training procedure in Section 3, including the Temporal-Aware Learning objective with quadratic temporal reweighting, later-step consistency regularization, and temporal target smoothing. Implementation details, including datasets, network architectures, and hyperparameter configurations, are provided in Appendix A.1 and Table 5. Training, evaluation, and ablation details are described in Section 4 and Appendix A.2–A.4. We will provide anonymized supplementary code containing the full TAL implementation, configurations, and scripts to reproduce the main results and ablation studies.

REFERENCES

Tobias Fischer Adam D. Hines, Michael Milford. A compact neuromorphic system for ultra–energy-efficient, on-device robot localization. *Science Robotics*, 10 103, 2025. doi: 10.1126/scirobotics.ads3968.

Filipp Akopyan, Jun Sawada, Andrew Cassidy, Rodrigo Alvarez-Icaza, John Arthur, Paul Merolla, Nabil Imam, Yutaka Nakamura, Pallab Datta, Gi-Joon Nam, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(10):1537–1557, 2015.

Srinivas Anumasa, Bhaskar Mukhoty, Velibor Bojkovic, Giulia De Masi, Huan Xiong, and Bin Gu. Enhancing training of spiking neural network with stochastic latency. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10900–10908, 2024.

Guo-qiang Bi and Mu-ming Poo. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18 (24):10464–10472, 1998.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359.

Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. In *International Conference on Learning Representations*, 2022.

Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. ieee, 2015.

Yiting Dong. Temporal knowledge sharing enable spiking neural network learning from past and future. *IEEE Transactions on Artificial Intelligence*, 5(7):3524–3534, 2024.

Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Advances in Neural Information Processing Systems*, 34:21056–21069, 2021a.

Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2661–2671, 2021b.

Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2020.

Yufei Guo, Yuanpei Chen, Liwen Zhang, Xiaode Liu, Yinglei Wang, Xuhui Huang, and Zhe Ma. Im-loss: information maximization loss for spiking neural networks. *Advances in Neural Information Processing Systems*, 35:156–166, 2022a.

Yufei Guo, Xinyi Tong, Yuanpei Chen, Liwen Zhang, Xiaode Liu, Zhe Ma, and Xuhui Huang. 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*, pp. 326–335, 2022b.

JiaKui Hu, Man Yao, Xuerui Qiu, Yuhong Chou, Yuxuan Cai, Ning Qiao, Yonghong Tian, Bo Xu, and Guoqi Li. High-performance temporal reversible spiking neural networks with o(l) training memory and o(1) inference cost. In *Forty-first International Conference on Machine Learning*, 2024a.

Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks toward deep residual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 36(2): 2353–2367, 2024b.

Youngeun Kim and Priyadarshini Panda. Revisiting batch normalization for training low-latency deep spiking neural networks from scratch. *Frontiers in Neuroscience*, 15:773954, 2021.

Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10:508, 2016.

Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in Neural Information Processing Systems*, 31, 2018.

Wei Liu, Li Yang, Mingxuan Zhao, Shuxun Wang, Jin Gao, Wenjuan Li, Bing Li, and Weiming Hu. Deeptage: Deep temporal-aligned gradient enhancement for optimizing spiking neural networks. In *The Thirteenth International Conference on Learning Representations*, 2025.

Xinhao Luo, Man Yao, Yuhong Chou, Bo Xu, and Guoqi Li. Integer-valued training and spike-driven inference spiking neural network for high-performance and energy-efficient object detection. In *European Conference on Computer Vision*, pp. 253–272. Springer, 2024.

Yongqiang Ma, Hao Wu, Mengjiao Zhu, Pengju Ren, Nanning Zheng, and Badong Chen. Reconstruction of visual image from functional magnetic resonance imaging using spiking neuron model. *IEEE Transactions on Cognitive and Developmental Systems*, 10(3):624–636, 2018. doi: 10.1109/TCDS.2017.2764948.

Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

Ruixin Mao, Lin Tang, Zihan Xia, Aoyu Shen, Haojie Wei, Zhaomin Zhang, Yuncheng Lu, Yu Long, Jinhong Guo, Liang Zhou, et al. Fsnap: An ultra-energy-efficient reconfigurable few-spikes-neuron-based snn processor supporting unified on-chip learning and adaptive time-window tuning. *IEEE Journal of Solid-State Circuits*, 2025.

Bhaskar Mukhoty, Velibor Bojkovic, William de Vazelhes, Xiaohan Zhao, Giulia De Masi, Huan Xiong, and Bin Gu. Direct training of snn using local zeroth order method. *Advances in Neural Information Processing Systems*, 36:18994–19014, 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. doi: 10.1109/MSP.2019.2931595.

Nicolas Perez-Nieves, Vincent CH Leung, Pier Luigi Dragotti, and Dan FM Goodman. Neural heterogeneity promotes robust learning. *Nature Communications*, 12(1):5791, 2021.

Xuerui Qiu, Rui-Jie Zhu, Yuhong Chou, Zhaorui Wang, Liang-jian Deng, and Guoqi Li. Gated attention coding for training high-performance and efficient spiking neural networks. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 38, pp. 601–610, 2024.

Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

Hangchi Shen, Qian Zheng, Huamin Wang, and Gang Pan. Rethinking the membrane dynamics and optimization objectives of spiking neural networks. *Advances in Neural Information Processing Systems*, 37:92697–92720, 2024.

SB Shrestha and GS Orchard. Spike layer error reassignment in time. *Advances in Neural Information Processing Systems*, 31, 2018.

Balachandran Swaminathan. *Neuromorphic Learning Systems for Resource-Aware, Adaptive Edge Intelligence*. The Pennsylvania State University, 2025.

Lihao Wang and Zhaofei Yu. Autaptic synaptic circuit enhances spatio-temporal predictive learning of spiking neural networks. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:331, 2018.

Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Zhouchen Lin. Online training through time for spiking neural networks. *Advances in Neural Information Processing Systems*, 35:20717–20730, 2022.

Jiaqi Yan, Changping Wang, De Ma, Huajin Tang, Qian Zheng, and Gang Pan. Training high performance spiking neural network by temporal model calibration. In *Forty-second International Conference on Machine Learning*, 2025.

Man Yao, Huanhuan Gao, Guangshe Zhao, Dingheng Wang, Yihan Lin, Zhaoxu Yang, and Guoqi Li. Temporal-wise attention spiking neural networks for event streams classification. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10201–10210, 2021. doi: 10.1109/ICCV48922.2021.01006.

Man Yao, Jiakui Hu, Guangshe Zhao, Yaoyuan Wang, Ziyang Zhang, Bo Xu, and Guoqi Li. Inherent redundancy in spiking neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16924–16934, 2023a.

Man Yao, Guangshe Zhao, Hengyu Zhang, Yifan Hu, Lei Deng, Yonghong Tian, Bo Xu, and Guoqi Li. Attention spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(8):9393–9410, 2023b.

Man Yao, Ole Richter, Guangshe Zhao, Ning Qiao, Yannan Xing, Dingheng Wang, Tianxiang Hu, Wei Fang, Tugba Demirci, Michele De Marchi, et al. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15(1):4464, 2024.

Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. Glif: A unified gated leaky integrate-and-fire neuron for spiking neural networks. *Advances in Neural Information Processing Systems*, 35: 32160–32171, 2022.

Chengting Yu, Zheming Gu, Da Li, Gaoang Wang, Aili Wang, and Erping Li. Stsc-snn: Spatio-temporal synaptic connection with temporal convolution and attention for spiking neural networks. *Frontiers in Neuroscience*, 16:1079357, 2022.

Kairong Yu, Tianqing Zhang, Hongwei Wang, and Qi Xu. Fsta-snn: Frequency-based spatial-temporal attention module for spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 22227–22235, 2025.

Shimin Zhang, Qu Yang, Chenxiang Ma, Jibin Wu, Haizhou Li, and Kay Chen Tan. Tc-lif: A two-compartment spiking neuron model for long-term sequential modelling. In *Proceedings of the AAAI conference on Artificial Intelligence*, volume 38, pp. 16838–16847, 2024.

Tianqing Zhang, Kairong Yu, Xian Zhong, Hongwei Wang, Qi Xu, and Qiang Zhang. Staa-snn: Spatial-temporal attention aggregator for spiking neural networks. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 13959–13969, 2025.

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.

Hanle Zheng, Zhong Zheng, Rui Hu, Bo Xiao, Yujie Wu, Fangwen Yu, Xue Liu, Guoqi Li, and Lei Deng. Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics. *Nature Communications*, 15(1):277, 2024.

Rui-Jie Zhu, Malu Zhang, Qihang Zhao, Haoyu Deng, Yule Duan, and Liang-Jian Deng. Tcja-snn: Temporal-channel joint attention for spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3):5112–5125, 2024.

A APPENDIX

A.1 EXPERIMENTAL SETTINGS

Dataset. We evaluate our method on widely used benchmark datasets, spanning both neuromorphic dynamic and static image tasks. For neuromorphic vision tasks, we use CIFAR10DVS dataset, which consists of 10,000 event streams converted from the original CIFAR10 images, simulating the outputs of a dynamic vision sensor. We adopt the standard split of 9,000 samples for training and 1,000 for testing. Additionally, we evaluate on DVSGesture, a dataset containing 1,464 samples of 11 distinct hand gestures recorded under varying illumination conditions, with the official partition of 1,176 training and 288 testing instances. N-Caltech101 dataset, with 8,831 samples spanning 101 object categories, is used with a standardized 9:1 train-test split. For traditional static image recognition, we employ several well-known datasets. CIFAR10 consists of 50,000 training and 10,000 test images across 10 object categories, while CIFAR100 includes 50,000 training and 10,000 test images across 100 fine-grained classes. For large-scale evaluation, we use ImageNet-1K dataset, which includes 1.2 million training images and 50,000 validation samples, covering 1,000 diverse object categories.

Network architecture. We adopt dataset-specific SNN backbones for different benchmarks. For CIFAR10-DVS, we employ a VGG11-based SNN with the architecture 64C3-LIF-128C3-LIF-AP2-256C3-LIF-256C3-LIF-AP2-512C3-LIF-512C3-LIF-AP2-512C3-LIF-512C3-LIF-AP2-10FC-LIF. Here, xCy denotes a convolutional layer with x output channels and a $y \times y$ kernel, MP/AP denotes max/average pooling, and nFC denotes a fully connected layer with n outputs. For DVSGesture, we use the architecture 128C3-LIF-MP2-128C3-LIF-MP2-128C3-LIF-MP2-128C3-LIF-MP2-128C3-LIF-MP2-0.5DP-512FC-LIF-0.5DP-100FC-LIF-Voting, where the voting layer performs one-dimensional average pooling over the classifier outputs. For N-Caltech101, the backbone is 64C3-LIF-MP2-128C3-LIF-MP2-256C3-LIF-MP2-256C3-LIF-MP2-512C3-LIF-0.8DP-1024FC-LIF-0.5DP-101FC-LIF. For static image classification, we adopt MS-ResNet18 for CIFAR-10/100 and MS-ResNet34 for ImageNet-1K.

Implementation details. All experiments are conducted using the PyTorch framework, leveraging NVIDIA Quadro RTX 8000 GPUs. Training settings and hyperparameters are detailed in Table 5.

Table 5: Hyper-parameter training settings for our TAL-SNNs on different datasets.

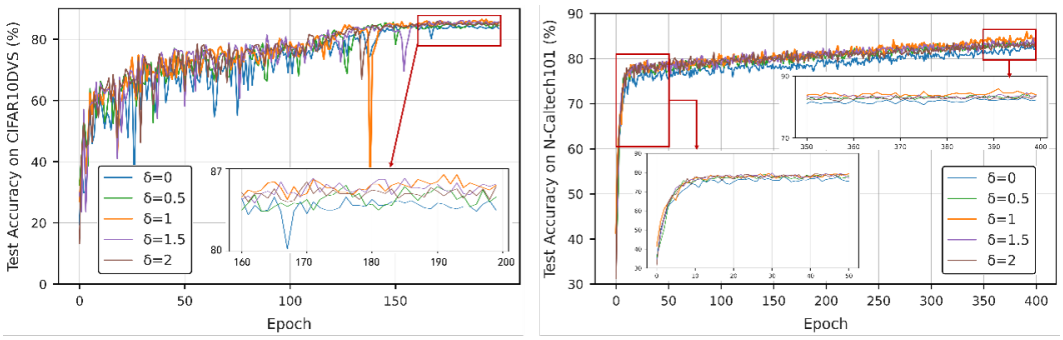
Dataset	Batch size	epoch	optimizer	Initial LR	Decay	Time Step	Mix up
CIFAR10DVS	32	200	SGD	0.1	5e-4	10/16	False
DVSGesture	32	500	Adam	0.001	CosALR	10/20	False
N-Caltech101	32	400	Adam	0.001	CosALR	10/14	True
CIFAR10	64	250	SGD	0.1	5e-5	4/6	True
CIFAR100	64	250	SGD	0.1	5e-5	4/6	True
ImageNet-1K	128	200	SGD	0.1	1e-5	4/6	True

A.2 HYPERPARAMETER SENSITIVITY ANALYSIS

The sensitivity analyses below are conducted under identical architectures and training settings. We focus on the temporal boundaries and regularization strength that directly control when and how strongly our method modifies supervision.

Effect of the offset δ. The offset parameter δ in quadratic temporal reweighting mechanism is introduced to preserve gradient flow from early time steps, ensuring that these steps do not receive zero weights and break the backpropagation path. Early in training, neurons accumulate limited information, and assigning zero weight to these early steps would suppress their gradients, potentially causing unstable convergence. A small positive δ ensures that every time step contributes a meaningful gradient signal, preventing sparse gradients from destabilizing the training process. As shown in Figure 10 and Table 6, this analysis underscores the importance of preserving gradient flow from early time steps, while still allowing later steps to dominate the optimization process.

Effect of t_{early}. The parameter t_{early} controls the temporal range over which temporal target smoothing is applied to early states. As shown in Table 7, the appropriate boundary depends on the total



(a) Test accuracy along with training epoch with different offset parameter δ on CIFAR10DVS.

(b) Test accuracy along with training epoch with different offset parameter δ on N-Caltech101.

Figure 10: Variation in test accuracy on CIFAR10DVS and N-Caltech101 datasets as epoch increases with different parameter δ.

Table 6: Effect of the offset parameter δ on accuracy over CIFAR10DVS and N-Caltech101.

Accuracy	Offset δ			
	0	1	1.5	2
CIFAR10DVS	84.6%	**86.5%**	86.2%	85.9%
N-Caltech101	83.20%	**85.93%**	84.63%	84.12%

temporal window. On CIFAR10DVS with $T = 10$, the accuracy increases from 86.3% at $t_{\text{early}} = 2$ to 86.8% at $t_{\text{early}} = 3$, but decreases to 84.9% when the smoothing range is further extended to $t_{\text{early}} = 4$. On CIFAR10 with $T = 6$, the best performance is achieved at $t_{\text{early}} = 2$, reaching 96.90%, whereas increasing the boundary to 3 and 4 reduces the accuracy to 94.21% and 91.53%, respectively. Interestingly, the preferred settings on both datasets correspond to approximately the first one-third of the temporal window. These results suggest that temporal target smoothing is most effective when restricted to relatively immature early states, while extending it into later, more reliable states may weaken useful supervision. Thus, t_{early} is better interpreted as a relative early-stage boundary rather than a dataset-specific absolute time step, with $t_{\text{early}} \approx T/3$ providing a simple practical reference.

Effect of t_{later}. The parameter t_{later} determines when later-step consistency regularization is activated. As shown in Table 8, performance improves substantially as the consistency constraint is shifted away from the rapidly evolving early states. On CIFAR10DVS with $T = 10$, the accuracy increases from 80.5% at $t_{\text{later}} = 2$ to 86.8% at $t_{\text{later}} = 5$, and remains unchanged for $t_{\text{later}} = 6$ and 7. A similar trend is observed on CIFAR10 with $T = 6$, where the accuracy increases from 90.28% at $t_{\text{later}} = 2$ to 96.90% at $t_{\text{later}} = 4$, and remains stable at $t_{\text{later}} = 5$. These results indicate that imposing consistency regularization too early may interfere with rapidly evolving temporal states, whereas once the regularization is restricted to sufficiently late states, TAL becomes relatively insensitive to the exact choice of t_{later}. Thus, t_{later} need not be tuned to a precise absolute step; choosing it around or after the midpoint of the temporal window provides a simple practical reference within the evaluated settings.

Effect of β. The coefficient β controls the strength of later-step consistency regularization in the overall objective. As shown in Table 9, a moderate consistency strength improves performance on both datasets, with the best results obtained at $\beta = 0.1$. On CIFAR10DVS, the accuracy increases from 86.5% at $\beta = 0$ to 86.8% at $\beta = 0.1$, while on CIFAR10 it improves from 96.71% to 96.90%. However, further increasing β leads to a clear performance degradation, with the accuracy dropping to 84.6% and 80.3% on CIFAR10DVS, and to 93.28% and 91.92% on CIFAR10, at $\beta = 0.2$ and 0.5, respectively. These results suggest that only a moderate consistency strength is required, while overly strong regularization may restrict useful variation among later temporal states.

Table 7: Sensitivity to the early-step boundary t_{early} of temporal target smoothing on CIFAR10DVS and CIFAR10.

Dataset	Early-step boundary t_{early}		
	2	3	4
CIFAR10DVS	86.3%	**86.8%**	84.9%
CIFAR10	**96.90%**	94.21%	91.53%

Table 8: Sensitivity to the starting step t_{later} of later-step consistency regularization on CIFAR10DVS and CIFAR10.

Accuracy	Later steps t_{later}					
	2	3	4	5	6	7
CIFAR10DVS	80.5 %	83.1%	84.2%	86.8%	86.8 %	86.8%
CIFAR10	90.28%	93.74%	96.90%	96.90%	-	-

A.3 Higher-order Temporal Weighting

To further examine whether the effectiveness of temporal reweighting depends on the specific quadratic form, we extend the weighting function to a general power-law family:

$$w_t^{(p)} = \left(\frac{t-1}{T-1}\right)^p + \delta, \qquad \hat{w}_t^{(p)} = \frac{w_t^{(p)}}{\sum_{k=1}^{T} w_k^{(p)}}, \tag{14}$$

where p controls how strongly the temporal weights are concentrated toward later states. Specifically, $p = 1$ corresponds to linear increasing weighting, $p = 2$ to the quadratic weighting adopted in our method, respectively. A larger p assigns relatively more emphasis to later temporal states while reducing the relative contribution of earlier ones.

As shown in Table 10, performance first improves as the power exponent increases from $p = 1$ to $p = 2$, but decreases when the weighting becomes more concentrated toward later states. On CIFAR10DVS, the accuracy increases from 85.9% with linear weighting ($p = 1$) to 86.5% with quadratic weighting ($p = 2$), and then decreases to 86.1% and 85.7% for $p = 3$ and $p = 4$, respectively. A similar trend is observed on CIFAR10, where the accuracy improves from 96.52% at $p = 1$ to 96.74% at $p = 2$, but drops to 96.16% and 95.37% as the exponent further increases. These results suggest that progressively increasing temporal emphasis is beneficial, but overly concentrating supervision on the latest states may reduce the contribution of useful earlier temporal information. The quadratic schedule therefore provides a favorable empirical balance between preserving early-step supervision and emphasizing later states with stronger local gradient consistency.

A.4 Temporal gradient consistency evolution during training

To investigate whether the observed temporal gradient consistency is stable throughout training rather than a phenomenon occurring only near convergence, we further analyze gradient relationships at different training epochs. Specifically, we compute the cosine similarity between gradients generated by different time steps at multiple checkpoints during training. Figure 11 presents the gradient similarity matrices at different training epochs. Although the absolute similarity values vary during optimization, the overall temporal pattern remains consistent: gradients from later time steps exhibit stronger mutual alignment, whereas early-to-late gradients show weaker consistency. This indicates that the temporal gradient evolution observed in our motivation analysis is not restricted to a specific training stage.

A.5 More case studies

A.5.1 Observation 2. Spatial spike activity evolves from diffuse to refined patterns.

We further visualize the spatial spike firing rate distribution of the penultimate layer at each time step via 2D maps by averaging spike firing rate of all channels. As shown in Figure 12, in early time steps,

Table 9: Sensitivity to the consistency regularization coefficient β on CIFAR10DVS and CIFAR10.

Accuracy	β				
	0	0.05	0.1	0.2	0.5
CIFAR10DVS	86.5%	86.7%	86.8%	84.6%	80.3%
CIFAR10	96.71%	96.83%	96.90%	93.28%	91.92%

Table 10: Effect of the power exponent p in increasing temporal weighting on CIFAR10DVS and CIFAR10.

Power exponent p	CIFAR10DVS	CIFAR10
1 (Linear)	85.9%	96.52%
2 (Quadratic)	86.5%	96.74%
3	86.1%	96.16%
4	85.7%	95.37%

spike activations are highly scattered and chaotic, containing almost no target-related structural information. As time progresses, spike activity gradually concentrates in specific spatial regions and eventually forms distinct class boundary contours. This evolutionary process indicates that useful task-relevant information accumulates incrementally rather than being uniformly distributed across all time steps.

A.5.2 Loss curve and convergence speed analysis

Figure 13 presents the corresponding training loss curves. Our method converges to a substantially lower loss value than both sequence-level supervision and time-step-specific supervision, reaching 4.2, whereas time-step-specific supervision converges to 7.0 and sequence-level supervision to 9.7. This result indicates more effective optimization. More importantly, the advantage of our method is not merely reflected in a lower final loss, but also in better gradient alignment across time steps, since our method reduces the conflicting gradient signals caused by unreliable early predictions. We further evaluate convergence efficiency by measuring how many epochs are required to reach a target accuracy. On CIFAR10DVS, our method reaches 80% accuracy after 95 epochs, while time-step-specific supervision requires 116 epochs and sequence-level supervision requires 140 epochs, corresponding to a convergence speedup of 18.1% and 32.1%, respectively. A similar trend is observed on N-Caltech101, where our method reaches the same accuracy threshold in 84 epochs, whereas time-step-specific supervision and sequence-level supervision require 163 and 186 epochs, respectively, yielding convergence speed improvements of 48.5% and 54.8%. Overall, these results show that our method not only improves the final performance, but also accelerates optimization by roughly 18% to 54% over common temporal loss functions.

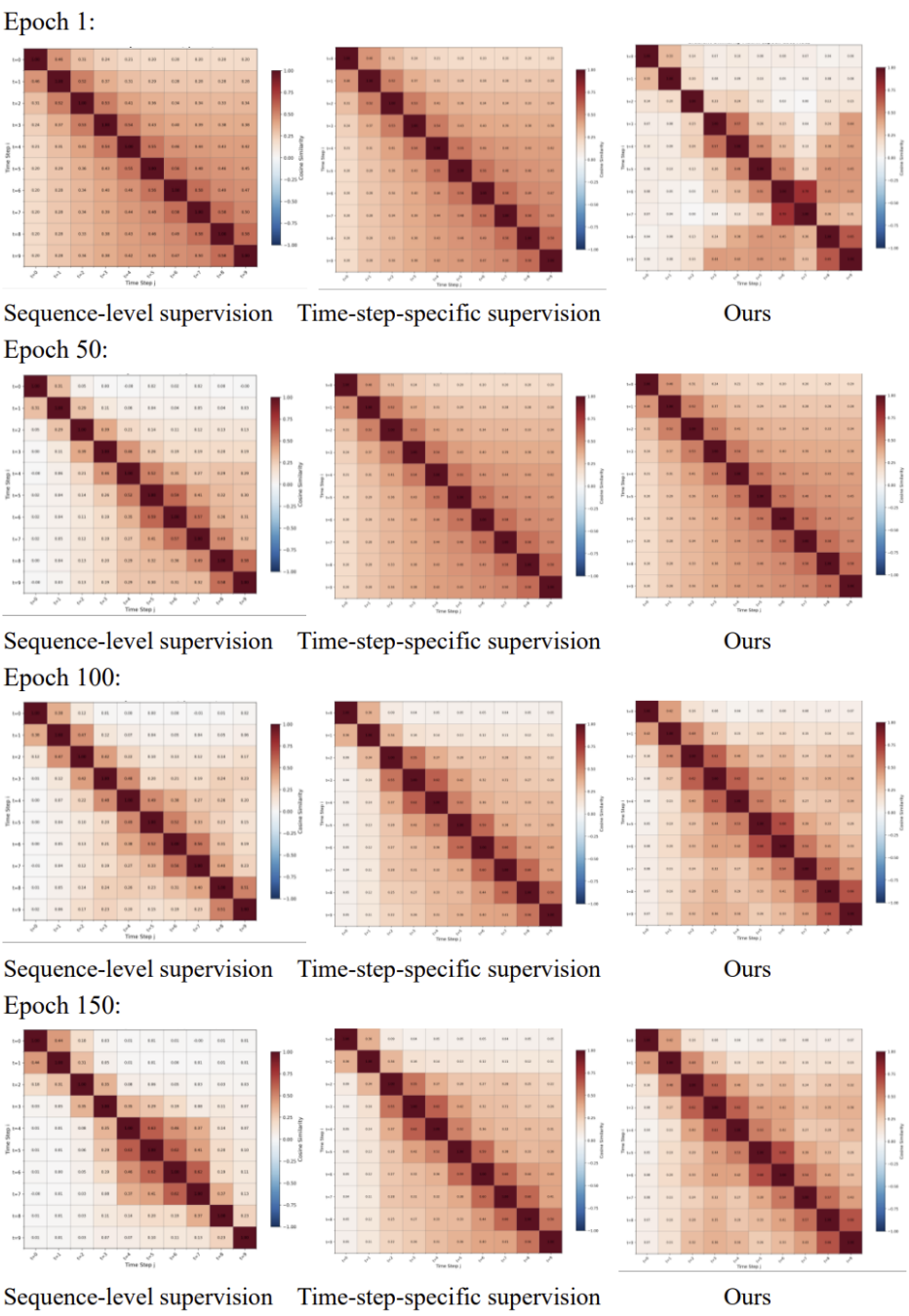
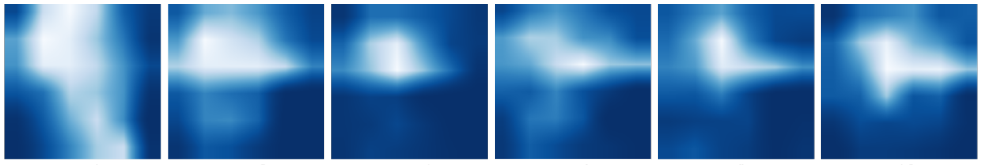
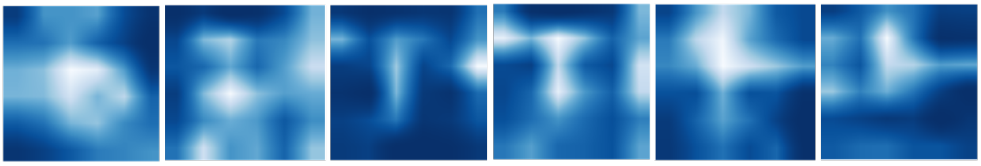


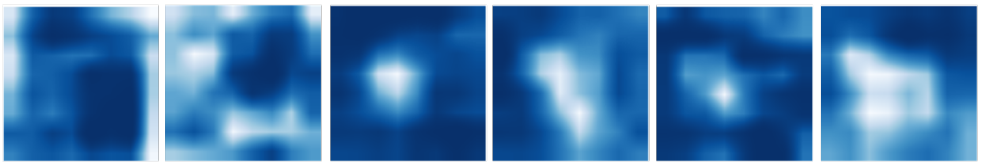
Figure 11: Gradient similarity matrices at different training epochs on CIFAR10DVS at $T = 10$.



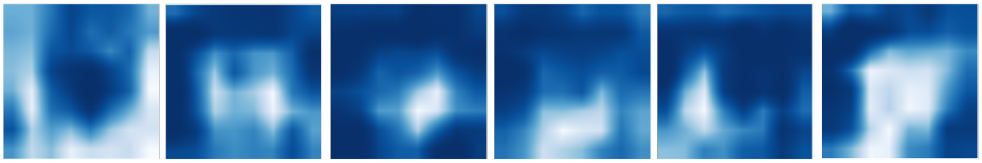
(a) Feature maps of the layer preceding the classifier at different time steps using sequence-level supervision on CIFAR10DVS.



(b) Feature maps of the layer preceding the classifier at different time steps using time-step-specific supervision on CIFAR10DVS.

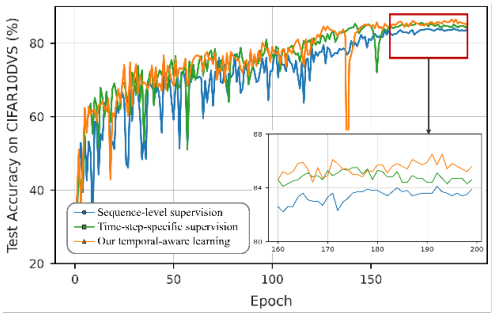


(c) Feature maps of the layer preceding the classifier at different time steps using sequence-level supervision on CIFAR10.

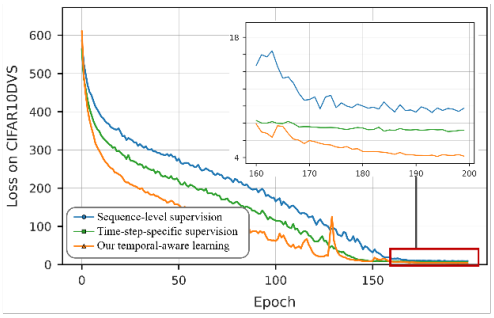


(d) Feature maps of the layer preceding the classifier at different time steps using time-step-specific supervision on CIFAR10.

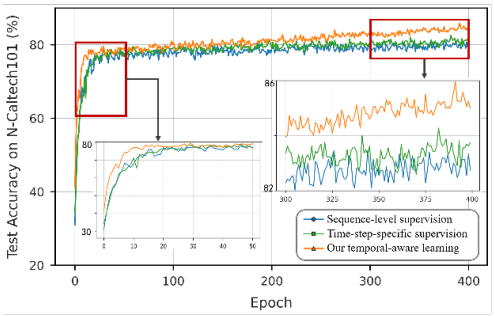
Figure 12: The evolution of intra-layer spike activity under different optimization objectives (sequence-level supervision and time-step-specific supervision) on CIFAR10DVS using VGGSNN and CIFAR10 using MS-ResNet18. Despite differences in objectives and tasks, both learning methods exhibit consistent behavior, where spatial firing patterns evolve from dispersed and unstructured activations to concentrated and structured representations over time.



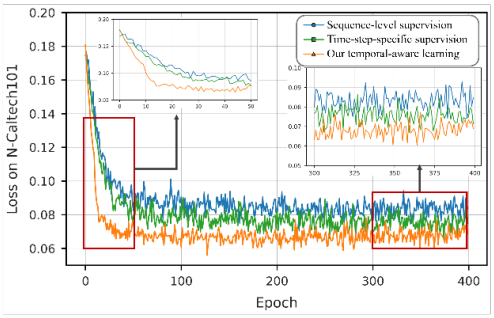
(a) Test accuracy along with training epoch using different loss functions on CIFAR10DVS.



(b) Loss along with training epoch using different loss functions on CIFAR10DVS.



(c) Test accuracy along with training epoch using different loss functions on N-Caltech101.



(d) Loss along with training epoch using different loss functions on N-Caltech101.

Figure 13: Variation in test accuracy and loss curve on CIFAR10DVS and N-Caltech101 datasets using different optimization objectives.