LRTB: LOW-RANK TEMPORAL BATCH NORMALIZATION IN SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Batch normalization (BN) facilitates the training of deep spiking neural networks (SNNs), but temporally evolving neuronal activations pose additional challenges for statistical estimation and parameter adaptation. Existing temporal normalization methods may overlook between-step mean drift when estimating cumulative variance, while independently learned time-specific affine parameters do not explicitly exploit temporal structure. To address these limitations, we propose Low-Rank Temporal Batch Normalization (LRTB), which combines exact cumulative statistical estimation with dynamics-inspired low-rank temporal parameterization. LRTB accounts for both within-step variability and between-step mean drift to accurately estimate the variance of accumulated activations. It further represents time-varying affine parameters using a compact set of exponential basis functions inspired by leaky integrate-and-fire dynamics, enabling structured temporal parameter sharing with fewer learnable affine parameters. Experiments on CIFAR-10, CIFAR-100, DVS-CIFAR10, and ImageNet demonstrate improvements over reported temporal normalization baselines.

1 INTRODUCTION

Spiking neural networks (SNNs) process information through discrete spikes and temporally evolving states, offering an energy-efficient alternative to conventional deep learning (Davies et al., 2018; Akopyan et al., 2015). Surrogate-gradient learning enables direct training of deep SNNs (Wu et al., 2018; Neftci et al., 2019; Fang et al., 2021a), where batch normalization (BN) plays a critical role in regulating activation scales. Unlike in ANNs, normalization in SNNs additionally modulates neuronal firing relative to the firing threshold (Zheng et al., 2021), raising a natural question: how should normalization accommodate the temporal dynamics inherent to spiking computation?

The membrane integration, leakage, spike generation, and reset of spiking neurons induce time-dependent activation distributions across simulation steps, motivating normalization strategies tailored to spiking computation (Zheng et al., 2021; Guo et al., 2023). Temporal normalization involves two fundamental design choices: how activation statistics are estimated across time and how time-varying transformations are parameterized. Existing methods have explored temporally shared, time-specific, and cumulative statistics (Kim & Panda, 2021; Duan et al., 2022; Jiang et al., 2024), together with various temporal parameter modulation schemes. Yet two aspects remain underexplored: the statistical fidelity of cumulative normalization and the structured representation of temporal affine parameters.

On the statistical side, cumulative normalization requires an accurate characterization of the activation distribution over the entire temporal window. Temporal Accumulated Batch Normalization (TAB) (Jiang et al., 2024) incorporates historical activation information by aggregating statistics over an expanding temporal window. However, its cumulative variance estimator averages the variances computed at individual time steps without accounting for differences among their means. Since the variance of the complete cumulative distribution depends on both within-step variability and temporal mean drift, averaging per-step variances alone does not, in general, recover the variance of all accumulated observations. This discrepancy motivates a statistically consistent estimation of cumulative variance that accounts for both sources of variability.

On the parameterization side, existing methods either employ time-shared affine parameters, which cannot adapt across simulation steps, or learn unconstrained time-specific parameters, which grow

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.

linearly with the simulation horizon and disregard the temporal structure of affine trajectories. An alternative is to represent these variations through a compact set of shared basis functions. In particular, the integration and leakage of LIF neurons give rise to exponentially decaying temporal dependencies (Burkitt, 2006), providing a natural inspiration for such a parameterization.

Our main contributions are summarized as follows:

- We introduce exact cumulative temporal statistics that account for both within-step activation variability and temporal mean drift, enabling accurate variance estimation over the complete cumulative activation distribution.
- We propose a dynamics-aligned low-rank parameterization that represents temporal affine trajectories using a compact set of exponential basis functions, enabling structured temporal parameter sharing with fewer learnable affine parameters.
- We evaluate LRTB on CIFAR-10, CIFAR-100 (Krizhevsky et al., 2009), DVS-CIFAR10 (Li et al., 2017), and ImageNet (Deng et al., 2009), demonstrating improvements over reported temporal normalization baselines across different architectures and simulation lengths.

2 Related Work

2.1 Normalization in Spiking Neural Networks

Batch normalization (BN) has played an important role in facilitating the training of deep neural networks (Ioffe & Szegedy, 2015; Santurkar et al., 2018). In SNNs, normalization must additionally account for neuronal firing behavior and temporally evolving activation distributions. Early work introduced NeuNorm to regulate neuronal input currents and improve the direct training of SNNs (Wu et al., 2019). Subsequent studies have further adapted normalization to the computational characteristics of spiking neurons. Threshold-dependent Batch Normalization (tdBN) incorporates the neuronal firing threshold into the normalization transformation, facilitating the training of deeper SNNs (Zheng et al., 2021). Postsynaptic Potential Batch Normalization (PSP-BN) employs the second raw moment instead of the conventional variance to regulate postsynaptic inputs (Ikegawa et al., 2022), whereas Membrane Potential Batch Normalization (MPBN) introduces normalization after membrane-potential updates and before spike generation (Guo et al., 2023). These approaches improve SNN training by adapting normalization to neuronal inputs, firing thresholds, and membrane-potential dynamics.

Beyond neuron-specific designs, temporal normalization methods have investigated how activation statistics and learnable parameters should be handled across simulation steps. Batch Normalization Through Time (BNTT) estimates separate statistics and learns time-specific scaling parameters to accommodate temporal variations in neuronal activity (Kim & Panda, 2021). Temporal Effective Batch Normalization (TEBN) instead employs temporally shared normalization statistics and affine parameters while introducing learnable time-dependent coefficients to modulate presynaptic inputs (Duan et al., 2022). Temporal Accumulated Batch Normalization (TAB) further incorporates historical activation information through cumulative statistics over an expanding temporal window, together with time-specific affine parameters and temporal importance weights (Jiang et al., 2024). These methods establish different strategies for temporal statistical estimation and parameter adaptation. Building upon cumulative normalization, LRTB focuses on the statistical fidelity of cumulative variance estimation and introduces a structured low-rank representation of temporal affine parameters, providing a complementary approach to temporal normalization in SNNs.

2.2 Dynamics-Inspired Temporal Representation

The integration and leakage dynamics of LIF neurons induce exponentially decaying temporal dependencies (Burkitt, 2006), while neuronal adaptation can operate over multiple characteristic timescales (Benda & Herz, 2003; La Camera et al., 2006). Related computational models introduce learnable membrane time constants (Fang et al., 2021b), and structured sequence models such as LMU and HiPPO represent temporal histories using compact basis expansions (Voelker et al., 2019; Gu et al., 2020). Unlike methods that modify neuronal dynamics or encode input histories, LRTB

uses a fixed exponential basis to represent time-varying normalization parameters, sharing temporal structure across channels while retaining channel-specific coefficients.

3 PRELIMINARY

3.1 SPIKING NEURON MODEL

Spiking neural networks (SNNs) process information through discrete spike trains. We adopt the discrete-time leaky integrate-and-fire (LIF) neuron with a hard-reset mechanism (Wu et al., 2018), whose dynamics are given by

$$u_i[t] = \lambda u_i[t-1](1 - o_i[t-1]) + I_i[t], \tag{1}$$

$$o_i[t] = \Theta(u_i[t] - V_{\text{th}}), \tag{2}$$

where $u_i[t]$ and $o_i[t] \in \{0, 1\}$ denote the membrane potential and output spike of neuron i at time step t, respectively. $I_i[t]$ is the synaptic input, V_{th} is the firing threshold, and $\Theta(\cdot)$ denotes the Heaviside step function. The leakage factor $\lambda \in (0, 1)$ governs the retention of membrane potential across time steps, giving rise to exponentially decaying temporal dynamics between reset events.

3.2 TEMPORAL BATCH NORMALIZATION IN SNNS

Batch Normalization (BN) normalizes neuronal activations using their estimated mean and variance, followed by a learnable affine transformation (Ioffe & Szegedy, 2015). For an activation $x[t]$ at time step t, the channel-wise transformation is given by

$$y[t] = \gamma \frac{x[t] - \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta, \tag{3}$$

where μ and σ^2 are the estimated mean and variance, γ and β are learnable affine parameters, and $\varepsilon > 0$ ensures numerical stability. In SNNs, membrane integration, leakage, and spike-induced resets lead to time-varying activation distributions, motivating normalization strategies tailored to temporal neuronal dynamics.

Existing SNN normalization methods differ in their estimation of temporal statistics and parameterization of temporal adaptation. Batch Normalization Through Time (BNTT) (Kim & Panda, 2021) estimates statistics and learns scaling parameters separately at each time step, whereas threshold-dependent BN (tdBN) (Zheng et al., 2021) employs temporally shared statistics and incorporates the firing threshold into normalization. Temporal Effective BN (TEBN) (Duan et al., 2022) combines temporally shared BN statistics with learnable time-specific modulation coefficients. Temporal Accumulated BN (TAB) (Jiang et al., 2024) instead aggregates statistics over the expanding temporal window $1{:}t$ and introduces time-specific affine parameters and temporal importance weights. Its channel-wise transformation is expressed as

$$y[t] = \omega[t] \left(\gamma[t] \frac{x[t] - \mu_{1:t}}{\sqrt{\widetilde{\sigma}^2_{1:t} + \varepsilon}} + \beta[t] \right), \tag{4}$$

where $\mu_{1:t}$ and $\widetilde{\sigma}^2_{1:t}$ denote the cumulative mean and estimated variance, respectively, while $\gamma[t]$, $\beta[t]$, and $\omega[t]$ are time-specific learnable parameters.

Despite these advances, two limitations remain. First, TAB estimates cumulative variance by averaging the variances of individual time steps, neglecting the contribution of temporal mean drift to the variance of the complete cumulative activation distribution. Second, existing methods either share affine parameters across time or introduce independently learned time-indexed parameters. The former cannot explicitly adapt the affine transformation to temporal variations, while the latter does not exploit the temporal structure associated with neuronal dynamics. These limitations motivate a temporal normalization framework that combines accurate cumulative statistics with structured temporal parameterization.

4 METHOD

LRTB combines the variance of the temporally pooled mini-batch activations with a low-rank parameterization of time-varying affine parameters.

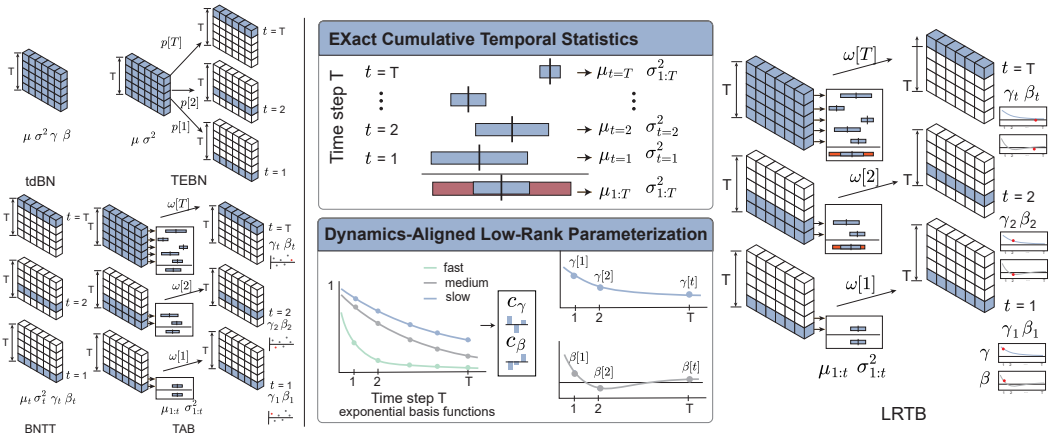

Figure 1: Comparison of normalization strategies in SNNs. Existing methods differ in their estimation of temporal statistics and parameterization of temporal adaptation. LRTB combines exact cumulative statistics with a dynamics-aligned low-rank parameterization of temporal affine trajectories.

4.1 Exact Cumulative Temporal Statistics

For an SNN simulated over T steps, let $x_n[t]$ denote the n-th activation of one channel at step t, where $n = 1, \ldots, N$ indexes batch and spatial positions. We assume the same N at each step and use the population-variance convention employed by BN:

$$\mu[t] = \frac{1}{N}\sum_{n=1}^{N} x_n[t], \qquad \sigma^2[t] = \frac{1}{N}\sum_{n=1}^{N} \left(x_n[t] - \mu[t]\right)^2. \tag{5}$$

The mean and variance of the tN observations pooled over steps $1{:}t$ are

$$\mu_{1:t} = \frac{1}{t}\sum_{s=1}^{t} \mu[s], \tag{6}$$

$$\sigma^2_{1:t} = \underbrace{\frac{1}{t}\sum_{s=1}^{t} \sigma^2[s]}_{\text{Within-step variance}} + \underbrace{\frac{1}{t}\sum_{s=1}^{t} \left(\mu[s] - \mu_{1:t}\right)^2}_{\text{Temporal mean-drift variance}}. \tag{7}$$

By the law of total variance, the cumulative variance comprises the average within-step variance and the additional variance induced by temporal mean drift. The latter is omitted when cumulative variance is estimated solely by averaging per-step variances, leading to an underestimation whenever the temporal means differ. By accounting for both components, the proposed estimator recovers the variance of the complete cumulative activation distribution.

The resulting cumulative statistics are used to normalize the activation at each time step:

$$z[t] = \frac{x[t] - \mu_{1:t}}{\sqrt{\sigma^2_{1:t} + \varepsilon}}. \tag{8}$$

Both moments admit constant-memory recursive updates (Appendix A). During inference, we first combine the stored per-step running moments with the same equations and then normalize causally; Appendix E states the required protocol.

4.2 Dynamics-Aligned Low-Rank Parameterization

While exact cumulative statistics characterize the evolving activation distribution, the affine transformation must also accommodate temporal variations in neuronal activity. Independently learning

affine parameters at each time step provides temporal flexibility but treats their trajectories as unconstrained sequences, without explicitly exploiting the temporal structure associated with neuronal dynamics. To address this limitation, we introduce a dynamics-aligned low-rank parameterization that represents temporal affine trajectories through a shared set of exponential basis functions.

Dynamics-aligned temporal basis. As described in Section 3, the LIF neuron exhibits exponentially decaying membrane dynamics between reset events. Motivated by this property, we construct a fixed temporal basis matrix $\mathbf{L}^{(T)} \in \mathbb{R}^{T \times r_{\text{eff}}}$, where $r_{\text{eff}} = \min(r, T)$ and r denotes the target basis size. Its elements are defined as

$$L_{t,k}^{(T)} = \exp\left(-\frac{t}{\tau_k(T)}\right), \quad \begin{matrix} t = 1, \dots, T, \\ k = 1, \dots, r_{\text{eff}}, \end{matrix} \tag{9}$$

where $\tau_k(T) > 0$ is the k-th basis time constant. We choose reference time constants on a logarithmic grid and rescale them with the simulation horizon; Appendix B gives the construction. The exponential form is a temporal inductive bias and does not assume that BN parameters reproduce membrane-potential dynamics.

Unless otherwise stated, LRTB uses $r_{\text{eff}} = \min(r, T)$, so that the number of basis functions does not exceed the temporal dimension. Here, r denotes the requested basis size, whereas r_{eff} denotes the number of basis functions actually used by the default model. In a separate overcomplete-basis ablation, we deliberately disable this truncation and use r basis functions even when $r > T$. This ablation is introduced only to examine whether restricting the basis size to at most T is beneficial.

Low-rank temporal affine parameters. Using the constructed basis, we parameterize the time-varying affine transformations as

$$\mathbf{\Gamma} = \mathbf{L}^{(T)} \mathbf{A}_\gamma, \qquad \mathbf{B} = \mathbf{L}^{(T)} \mathbf{A}_\beta, \tag{10}$$

where $\mathbf{\Gamma}, \mathbf{B} \in \mathbb{R}^{T \times C}$ contain the scaling and shifting parameters of all C channels across T time steps, respectively, and $\mathbf{A}_\gamma, \mathbf{A}_\beta \in \mathbb{R}^{r_{\text{eff}} \times C}$ are learnable coefficient matrices. Accordingly, the affine parameters of channel c at time step t are obtained as

$$\gamma[t] = \sum_{k=1}^{r_{\text{eff}}} L_{t,k}^{(T)} a_{\gamma,k}, \qquad \beta[t] = \sum_{k=1}^{r_{\text{eff}}} L_{t,k}^{(T)} a_{\beta,k}. \tag{11}$$

This formulation constrains the temporal affine trajectories to the subspace spanned by the exponential basis, with $\text{rank}(\mathbf{\Gamma}), \text{rank}(\mathbf{B}) \leq r_{\text{eff}}$. By sharing the temporal basis across channels while learning channel-specific coefficients, LRTB captures temporal variations through a compact, structured representation. For $r \ll T$, the number of learnable affine parameters is reduced from $2T$ to $2r$, while retaining time-dependent affine transformations.

LRTB transformation. Combining the exact cumulative statistics with the low-rank temporal affine parameters, the proposed LRTB transformation is formulated as

$$\begin{aligned} y[t] &= \omega[t] \left(\gamma[t] z[t] + \beta[t]\right) \\ &= \omega[t] \sum_{k=1}^{r_{\text{eff}}} L_{t,k}^{(T)} \left(a_{\gamma,k} z[t] + a_{\beta,k}\right), \end{aligned} \tag{12}$$

where $z[t]$ denotes the normalized activation obtained from Eq. equation 8.

Following TAB, we retain a learnable, temporal importance weight $\omega[t]$ to modulate the contributions of individual time steps. Unlike the affine trajectories, these weights remain independently learnable, providing additional temporal flexibility beyond the fixed exponential basis.

The temporal basis $\mathbf{L}^{(T)}$ is fixed, whereas the coefficient matrices $\mathbf{A}_\gamma$ and $\mathbf{A}_\beta$, together with the temporal importance weights, are optimized during training. For $r \leq T$, LRTB requires $2rC + T$ learnable normalization parameters, combining exact cumulative statistical estimation with structured temporal affine adaptation.

5 Experiments

We evaluate LRTB on static image classification and neuromorphic benchmarks, including CIFAR-10, CIFAR-100, ImageNet, and DVS-CIFAR10. Beyond comparisons with existing SNN normal-

Table 1: Performance comparison with existing SNN methods on CIFAR-10, CIFAR-100, DVS-CIFAR10, and ImageNet. Results of competing methods are taken from their original publications. Within each dataset, comparisons using identical architectures and simulation lengths are emphasized. Bold entries indicate our results.

Dataset	Method	Training	Architecture	Time Steps	Accuracy (%)
CIFAR-10	SPIKE-NORM (Sengupta et al., 2019)	ANN-to-SNN	VGG-16	2500	91.55
	NeuNorm (Wu et al., 2019)	Surrogate Gradient	CIFARNet	12	90.53
	BNTT (Kim & Panda, 2021)	Surrogate Gradient	VGG-9	20	90.30
	tdBN (Zheng et al., 2021)	Surrogate Gradient	ResNet-19	4	92.92
	TEBN (Duan et al., 2022)	Surrogate Gradient	VGG-9	4	92.81
	TEBN (Duan et al., 2022)	Surrogate Gradient	ResNet-19	4	94.70
	TAB (Jiang et al., 2024)	Surrogate Gradient	VGG-9	4	93.41
	TAB (Jiang et al., 2024)	Surrogate Gradient	ResNet-19	4	94.76
	LRTB (Ours)	Surrogate Gradient	VGG-9	4	**94.08**
	LRTB (Ours)	Surrogate Gradient	ResNet-19	4	**96.13**
CIFAR-100	SPIKE-NORM (Sengupta et al., 2019)	ANN-to-SNN	VGG-16	2500	70.90
	BNTT (Kim & Panda, 2021)	Surrogate Gradient	VGG-11	50	66.60
	TEBN (Duan et al., 2022)	Surrogate Gradient	VGG-11	4	74.37
	TEBN (Duan et al., 2022)	Surrogate Gradient	ResNet-19	4	76.13
	TAB (Jiang et al., 2024)	Surrogate Gradient	VGG-11	4	75.89
	TAB (Jiang et al., 2024)	Surrogate Gradient	ResNet-19	4	76.81
	LRTB (Ours)	Surrogate Gradient	VGG-11	4	**76.31**
	LRTB (Ours)	Surrogate Gradient	ResNet-19	4	**78.06**
DVS-CIFAR10	NeuNorm (Wu et al., 2019)	Surrogate Gradient	7-layer CNN	40	60.50
	BNTT (Kim & Panda, 2021)	Surrogate Gradient	7-layer CNN	20	63.20
	tdBN (Zheng et al., 2021)	Surrogate Gradient	ResNet-19	10	67.80
	TEBN (Duan et al., 2022)	Surrogate Gradient	7-layer CNN	10	75.10
	TAB (Jiang et al., 2024)	Surrogate Gradient	7-layer CNN	4	76.70
	LRTB (Ours)	Surrogate Gradient	7-layer CNN	4	**77.60**
ImageNet	SlipReLU (Jiang et al., 2023)	ANN-to-SNN	ResNet-34	32	66.61
	tdBN (Zheng et al., 2021)	Surrogate Gradient	ResNet-34	6	63.72
	TEBN (Duan et al., 2022)	Surrogate Gradient	ResNet-34	4	64.29
	TAB (Jiang et al., 2024)	Surrogate Gradient	ResNet-34	4	67.78
	LRTB (Ours)	Surrogate Gradient	ResNet-34	4	**68.75**

Table 2: Ablation and design analysis of LRTB on CIFAR-10 with $T = 4$. The component, basis, rank, and basis-variant ablations use ResNet-19, whereas the temporal-weighting ablation uses VGG-9.

(a) Component Ablation

Method	Exact Statistics	Low-rank Parameterization	Accuracy (%)
TAB	×	×	94.76
+ Exact Statistics	✓	×	95.66
+ Low-rank Parameterization	×	✓	95.78
LRTB	✓	✓	**96.13**

(b) Temporal Parameterization Analysis

Temporal Basis

Basis	Acc. (%)
Exponential (Ours)	**96.13**
DCT	95.76
Fourier	95.67
Polynomial	95.67
Wavelet	95.91
Learnable	95.69

Temporal Rank

Rank r	Acc. (%)
1	95.50
2	96.04
3	**96.13**
4	95.87
8	95.73

Additional Design Choices

Basis Variants	
Exponential basis	**96.13**
+ Constant basis	95.78
Expanded decay range	95.80
Temporal Weighting	
LRTB without $\omega[t]$	93.66
LRTB with $\omega[t]$	94.08

ization methods, we investigate the effects of temporal basis selection and rank, and analyze the low-rank structure of temporal affine parameters learned by an unconstrained normalization method.

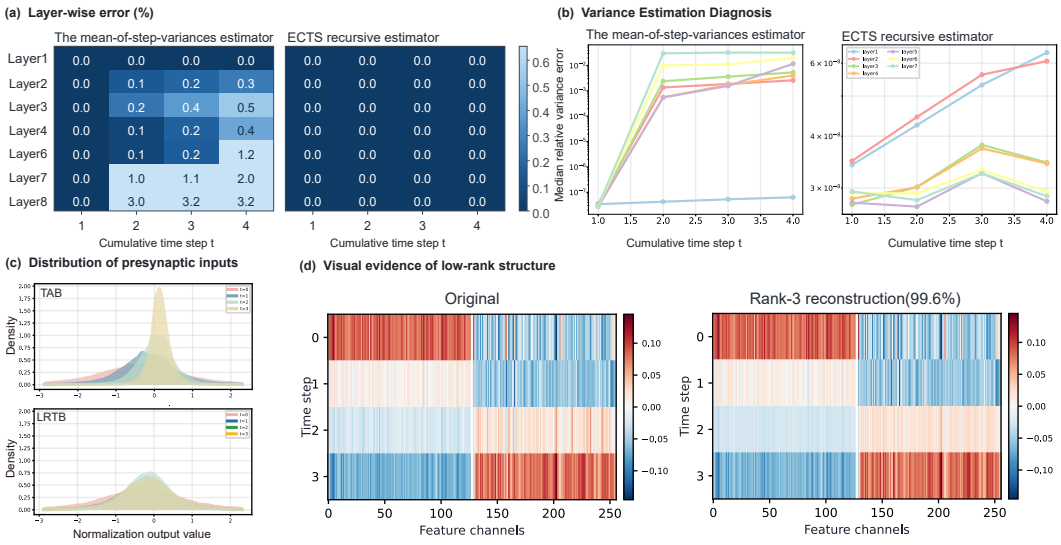

Figure 2: Mechanism analysis. (a,b) Relative error of the mean-of-step-variances estimator and the proposed pooled-variance estimator. (c) Temporal distributions of normalization outputs. (d) A temporally centered TAB affine matrix and its rank-3 reconstruction. Error aggregation and projection metrics are defined in Appendix C.

5.1 Comparison with Existing Methods

We first evaluate LRTB against existing SNN methods across static and neuromorphic benchmarks. Table 1 summarizes the classification performance under different network architectures and simulation lengths.

Static image classification. On CIFAR-10 at $T = 4$, LRTB achieves 94.08% with VGG-9 and 96.13% with ResNet-19(Sengupta et al., 2019), improving on the corresponding TAB results by 0.67 and 1.37 percentage points, respectively.On CIFAR-100 at $T = 4$, LRTB improves VGG-11 accuracy from 75.89% to 76.31% and ResNet-19 accuracy from 76.81% to 78.06%, gains of 0.42 and 1.25 percentage points over TAB.

We additionally evaluate LRTB at $T = 2$ and $T = 6$. LRTB consistently improves over the corresponding same-horizon baseline. Complete results are provided in Appendix F.

Neuromorphic classification. On DVS-CIFAR10, LRTB achieves 77.60% accuracy with a 7-layer CNN at $T = 4$, compared with 76.70% for TAB under the same architecture and simulation length. This result indicates that the improvement achieved by LRTB also extends to the evaluated event-based recognition setting.

Large-scale image classification. We further evaluate LRTB on ImageNet using ResNet-34. As shown in Table 1, LRTB achieves 68.75% top-1 accuracy at $T = 4$, improving upon TAB by 0.97 percentage points under the same architecture and simulation length. This result demonstrates that the performance improvement of LRTB is also observed on a large-scale image classification benchmark.

5.2 Component Ablation

Figure 2(a,b) quantifies the resulting bias. The mean-of-step-variances estimator omits the between-step variance induced by temporal mean drift, systematically underestimating the true cumulative variance. The omitted fraction is negligible at shallow layers but grows consistently with depth and with the cumulative time step t: at the deepest layer it reaches a median of 3.2% and a 95th-percentile of 8.8% over batch and channel series, corresponding to a systematic distortion of the normalized activation scale (Figure 2(c), std inflation up to 1.05$\times$ at the 95th percentile). In contrast,

the proposed exact recursive estimator remains within numerical precision ($< 10^{-6}$ relative error) at all layers and time steps.

To isolate the contributions of exact cumulative statistics and low-rank affine parameterization, we conduct an ablation study on CIFAR-10 using ResNet-19 with $T = 4$. Starting from TAB, we enable each component separately and then jointly while keeping all other settings fixed 2(a).

5.3 Temporal Basis and Rank Analysis

We next investigate the two key design choices of the proposed low-rank temporal parameterization: the functional form of the temporal basis and the number of basis functions. Unless otherwise specified, experiments in this subsection are conducted on CIFAR-10 using ResNet-19 with $T = 4$.

Comparison with alternative basis functions. To investigate the effect of temporal basis selection, we compare the proposed exponential basis with several alternatives, including discrete cosine transform (DCT), Fourier, polynomial, wavelet, and learnable bases. All other components of LRTB are kept unchanged. The results are summarized in the Temporal Basis block of Table 2(b).

The exponential basis achieves 96.13% accuracy, outperforming all evaluated alternatives. In particular, it improves upon the wavelet and DCT bases by 0.22 and 0.37 percentage points, respectively. The learnable basis achieves 95.69% accuracy, indicating that introducing additional flexibility in the temporal basis does not necessarily improve classification performance.

These results suggest that the exponential basis provides an effective temporal inductive bias for modeling time-varying affine parameters in SNNs. Its functional form is consistent with the exponential leakage dynamics of LIF neurons, offering a structured alternative to generic temporal basis functions.

Effect of temporal rank. We further investigate the effect of the temporal rank r while keeping the network architecture, simulation length, and other training settings unchanged. The results are reported in the Temporal Rank block of Table 2(b).

Increasing the temporal rank from 1 to 3 improves the classification accuracy from 95.50% to 96.13%, indicating that multiple temporal components are beneficial for representing time-varying affine parameters. The highest accuracy is obtained at $r = 3$, whereas increasing the effective rank to 4 yields a slightly lower accuracy of 95.87%.

These results suggest that a compact temporal basis can provide sufficient representational capacity for the evaluated setting.

Notably, we also evaluate the case where the rank exceeds the simulation length, i.e., $r = 8$ with $T = 4$, without the truncation $r_{\text{eff}} = \min(r, T)$, and the accuracy of 95.73% remains comparable to that of $r = 4$, confirming that redundant basis functions beyond the temporal dimension offer no additional benefit. These results indicate that a compact temporal basis provides sufficient representational capacity, and we therefore adopt $r = 3$ as the default.

5.4 Analysis of Temporal Affine Trajectories

A central motivation of LRTB is that temporal affine parameters can be represented using a compact set of shared basis functions, rather than being learned independently at every time step. To examine this premise, we analyze the temporal affine parameters learned by the unconstrained TAB model.

Specifically, we project the learned scaling and shifting parameters onto the predefined exponential basis and measure the fraction of parameter energy retained after projection. This analysis evaluates how accurately the temporal trajectories learned by TAB can be represented within the subspace adopted by LRTB.

The rank-2 projection retains 96.41% of the parameter energy on average under the balanced metric. Increasing the rank to 3 further improves the reconstruction, retaining 99.6% of the joint parameter energy and 97.99% under the balanced metric. Across the evaluated layers, the reported retained energy is no lower than 95.24%.

These results indicate that the temporal affine trajectories learned by TAB can be represented with high fidelity using the proposed exponential basis in the evaluated setting. Together with the rank

ablation in Section 5.3, this observation provides empirical support for representing time-varying affine parameters using a compact, structured temporal basis.

The complete projection analysis, including layer-wise results and the balanced evaluation protocol, is provided in Appendix C.

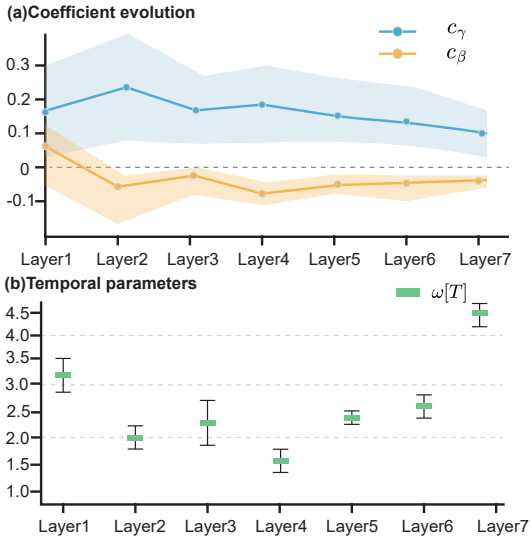

Figure 3: Layer-wise temporal profiles of the learned LRTB parameters on CIFAR-10 using VGG-9 with $T = 4$.

5.5 Temporal Parameterization Analysis

Constant and slow-decay basis variants. Although the proposed exponential basis does not contain an explicit constant component, slowly varying trajectories can be approximated by exponential functions with sufficiently large time constants. We therefore evaluate two additional variants: augmenting the proposed basis with an all-ones column and expanding the decay range by setting $\tau_{\max} = 10T$. As shown in the Basis Variants block of Table 2(b), the constant-augmented and expanded-range variants achieve 95.78% and 95.80% accuracy, respectively, compared with 96.13% for the proposed exponential basis. These results indicate that neither an explicit constant component nor an additional slow-decay component improves performance under the evaluated setting. Appendix B.1 discusses the relationship between constant trajectories and exponential basis functions in more detail.

Temporal importance weighting. We further examine the temporal importance weight $\omega[t]$ used in Eq. equation 12. Because the corresponding TAB comparison in Table 1 uses VGG-9, this ablation is conducted on CIFAR-10 with VGG-9 and $T = 4$. As reported in the Temporal Weighting block of Table 2(b), removing $\omega[t]$ reduces the accuracy from 94.08% to 93.66%, a decrease of 0.42 percentage points. This result supports the practical utility of the temporal gate when the affine trajectories are constrained to a low-rank basis. Appendix B.2 explains why $\omega[t]$ cannot generally be absorbed into the coefficient matrices under this constraint.

Learned temporal parameter profiles. To examine how LRTB adapts normalization across network depth and simulation time, Figure 3 visualizes the learned parameters of each normalization layer. For the affine scale and bias, we report the channel-wise averages $\bar{\gamma}_\ell[t] = C_\ell^{-1} \sum \gamma_\ell[t]$ and $\bar{\beta}_\ell[t] = C_\ell^{-1} \sum \beta_\ell[t]$. The temporal weight $\omega_\ell[t]$ is shown directly.

6 Conclusion

In this paper, we proposed Low-Rank Temporal Batch Normalization (LRTB) for spiking neural networks, addressing two complementary aspects of temporal normalization: the statistical fidelity of cumulative estimates and the structured parameterization of time-varying affine transformations. LRTB accounts for temporal mean drift to obtain exact cumulative variance and employs a compact set of exponential basis functions inspired by neuronal dynamics to enable temporal parameter sharing. Experiments on CIFAR-10, CIFAR-100, DVS-CIFAR10, and ImageNet demonstrate improvements over reported temporal normalization baselines, while analyses of temporal basis selection, rank, and affine parameter trajectories support the proposed parameterization. These findings highlight the importance of jointly considering accurate temporal statistics and structured temporal adaptation when designing normalization methods for SNNs.

AI USE STATEMENT

Generative AI tools were used for language editing and manuscript organization. The authors independently verified all technical claims, derivations, experiments, citations, and reported results and take full responsibility for the manuscript.

REPRODUCIBILITY STATEMENT

Detailed experimental configurations, including implementation and training settings, are provided in the Appendix. The complete experimental code necessary to reproduce the results is included in the supplementary material submitted with this paper.

REFERENCES

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.

Jan Benda and Andreas VM Herz. A universal model for spike-frequency adaptation. *Neural computation*, 15(11):2523–2564, 2003.

Léon Bottou. Stochastic gradient descent tricks. In *Neural networks: tricks of the trade: second edition*, pp. 421–436. Springer, 2012.

Anthony N Burkitt. A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biological cybernetics*, 95(1):1–19, 2006.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee micro*, 38(1):82–99, 2018.

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.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Chaoteng Duan, Jianhao Ding, Shiyan Chen, Zhaofei Yu, and Tiejun Huang. Temporal effective batch normalization in spiking neural networks. *Advances in Neural Information Processing Systems*, 35:34377–34390, 2022.

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 *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2641–2651. IEEE, 2021b.

Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480, 2023.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33: 1474–1487, 2020.

Yufei Guo, Yuhan Zhang, Yuanpei Chen, Weihang Peng, Xiaode Liu, Liwen Zhang, Xuhui Huang, and Zhe Ma. Membrane potential batch normalization for spiking neural networks. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19363–19373. IEEE, 2023.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Shin-ichi Ikegawa, Ryuji Saiin, Yoshihide Sawada, and Naotake Natori. Rethinking the role of normalization and residual blocks for spiking neural networks. *Sensors*, 22(8):2876, 2022.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pp. 448–456. pmlr, 2015.

Haiyan Jiang, Srinivas Anumasa, Giulia De Masi, Huan Xiong, and Bin Gu. A unified optimization framework of ann-snn conversion: towards optimal mapping from activation values to firing rates. In *International conference on machine learning*, pp. 14945–14974. PMLR, 2023.

Haiyan Jiang, Vincent Zoonekynd, Giulia De Masi, Bin Gu, and Huan Xiong. Tab: Temporal accumulated batch normalization in spiking neural networks. In *International Conference on Learning Representations*, volume 2024, pp. 6890–6916, 2024.

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

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Giancarlo La Camera, Alexander Rauch, David Thurbon, Hans-R Luscher, Walter Senn, and Stefano Fusi. Multiple time scales of temporal response in pyramidal and fast spiking cortical neurons. *Journal of neurophysiology*, 96(6):3448–3464, 2006.

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.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

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.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? *Advances in neural information processing systems*, 31, 2018.

Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in neuroscience*, 13:95, 2019.

Aaron Voelker, Ivana Kajić, and Chris Eliasmith. Legendre memory units: Continuous-time representation in recurrent neural networks. *Advances in neural information processing systems*, 32, 2019.

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.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1311–1318, 2019.

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.

APPENDIX

A EXACT CUMULATIVE STATISTICS

Let S be uniformly distributed on $\{1, \dots, t\}$ and let X be an activation drawn uniformly from the batch and spatial positions at step S. With $\mathbb{E}[X \mid S = s] = \mu[s]$ and $\mathrm{Var}(X \mid S = s) = \sigma^2[s]$, the laws of total expectation and total variance give

$$\mu_{1:t} = \mathbb{E}[X] = \frac{1}{t}\sum_{s=1}^{t} \mu[s], \tag{13}$$

$$\sigma^2_{1:t} = \mathbb{E}[\mathrm{Var}(X \mid S)] + \mathrm{Var}(\mathbb{E}[X \mid S]) \tag{14}$$

$$= \frac{1}{t}\sum_{s=1}^{t} \sigma^2[s] + \frac{1}{t}\sum_{s=1}^{t} \big(\mu[s] - \mu_{1:t}\big)^2. \tag{15}$$

This identity is exact for the tN activations pooled over time when every step contains the same number N of observations and each step variance uses divisor N. It does not claim to recover the unknown population variance from a finite mini-batch.

Stable online update. Let $\bar{\mu}_t$ be the cumulative mean, $\bar{v}_t$ the average within-step variance, and D_t the sum of squared deviations of the step means. Initialize all three quantities to zero. For $t \geq 1$,

$$\delta_t = \mu[t] - \bar{\mu}_{t-1}, \tag{16}$$

$$\bar{\mu}_t = \bar{\mu}_{t-1} + \delta_t / t, \tag{17}$$

$$\bar{v}_t = \bar{v}_{t-1} + \big(\sigma^2[t] - \bar{v}_{t-1}\big)/t, \tag{18}$$

$$D_t = D_{t-1} + \delta_t\big(\mu[t] - \bar{\mu}_t\big), \tag{19}$$

$$\sigma^2_{1:t} = \bar{v}_t + D_t/t. \tag{20}$$

The update is a Welford-style computation and requires constant memory per channel. It is preferable to computing second moments by subtraction, particularly in mixed precision. Moment accumulation should be performed in FP32 even when the network uses lower-precision activations.

Inference. During training, maintain running estimates m_s and v_s of the per-step mean and population variance for each channel. At inference, combine $\{m_s, v_s\}_{s=1}^{t}$ with the equations above and do not use statistics from the current test batch. This distinction is essential: recomputing batch moments at test time changes the evaluation protocol and leaks information across samples.

B BASIS CONSTRUCTION, HORIZON SCALING, AND INITIALIZATION

We construct the temporal basis at a reference simulation horizon $T_0 = 4$. The leak factor used to construct the basis is $\lambda_{\mathrm{LIF}} = 0.9$, which is identical to the leak factor of the LIF neurons used in our experiments. Its corresponding time constant, measured in simulation steps, is

$$\tau_{\mathrm{LIF}} = -\frac{1}{\log \lambda_{\mathrm{LIF}}} \approx 10. \tag{21}$$

At a reference horizon $T_0 = 4$, we use r_{eff} time constants logarithmically spaced from one step to τ_{LIF}:

$$\tau_k^{(0)} = \tau_{\mathrm{LIF}}^{(k-1)/(r_{\mathrm{eff}}-1)}, \qquad k = 1, \dots, r_{\mathrm{eff}}. \tag{22}$$

For $r_{\mathrm{eff}} = 1$, we set $\tau_1^{(0)} = \tau_{\mathrm{LIF}}$. Equation equation 22 assumes that one simulation step is the desired lower endpoint.

To preserve the basis profile as a function of normalized time $s = t/T$ when changing the horizon from T_0 to T, set

$$\tau_k(T) = \tau_k^{(0)} \frac{T}{T_0}, \qquad \lambda_k(T) = \lambda_k(T_0)^{T_0/T}. \tag{23}$$

Then $\exp[-t/\tau_k(T)] = \exp[-sT_0/\tau_k^{(0)}]$. This scaling changes only the normalization basis; it does not alter the intrinsic leak of the SNN neuron.

B.1 Constant-Mode Representation in the Exponential Basis

The proposed basis does not include an explicit constant column. This does not prevent it from representing slowly varying or approximately constant trajectories over a finite simulation window. In particular, a constant function is the limiting case of an exponential basis function:

$$\lim_{\tau \to \infty} \exp\left(-\frac{t}{\tau}\right) = 1. \tag{24}$$

Accordingly, increasing the largest time constant produces a progressively slower basis component whose values remain closer to one throughout the simulation horizon. Linear combinations of the available exponential components can also approximate a constant-like trajectory without requiring a dedicated constant mode.

The corresponding empirical comparison is reported in the Basis Variants block of Table 2(b). The results show that neither adding an exact constant column nor extending the basis with a slower-decaying component improves accuracy in the evaluated setting. One possible explanation is that very slow components introduce nearly redundant directions over the short simulation horizon and weaken the intended temporal inductive bias. This interpretation is a hypothesis rather than a direct consequence of Eq. equation 24.

B.2 Role of the Temporal Importance Weight $\omega[t]$

One may ask whether the temporal importance weight $\omega[t]$ is redundant, because it multiplies both affine terms in

$$y_[t] = \omega[t]\big(\gamma[t]z[t] + \beta[t]\big). \tag{25}$$

If $\gamma[t]$ and $\beta[t]$ were independently learnable at every time step, the answer would be yes: one could define $\gamma'[t] = \omega[t]\gamma[t]$ and $\beta'[t] = \omega[t]\beta[t]$ without changing the function. This is the case for an unconstrained time-specific affine parameterization.

The situation differs under the low-rank constraint used by LRTB. Substituting Eq. equation 10 into Eq. equation 25 gives the effective affine trajectories

$$\mathbf{\Gamma}_{\text{eff}} = \operatorname{diag}(\boldsymbol{\omega})\mathbf{L}^{(T)}\mathbf{A}_\gamma, \qquad \mathbf{B}_{\text{eff}} = \operatorname{diag}(\boldsymbol{\omega})\mathbf{L}^{(T)}\mathbf{A}_\beta. \tag{26}$$

Without $\boldsymbol{\omega}$, every channel trajectory is restricted to the fixed column space $\operatorname{col}(\mathbf{L}^{(T)})$. A freely learned positive $\boldsymbol{\omega}$ applies the same time-dependent row scaling to all channels, yielding the modulated space $\operatorname{col}(\operatorname{diag}(\boldsymbol{\omega})\mathbf{L}^{(T)})$. In general, this row scaling cannot be absorbed into $\mathbf{A}_\gamma$ and $\mathbf{A}_\beta$ unless it preserves the original basis subspace. Thus, for $r_{\text{eff}} < T$, $\omega[t]$ provides a parameter-efficient temporal gate: it adjusts the global importance of each step using only T channel-shared parameters, while the low-rank coefficients retain channel-specific scale and bias structure.

The corresponding empirical ablation is reported in the Temporal Weighting block of Table 2(b). The observed improvement should be interpreted as evidence for the practical utility of $\omega[t]$ under the evaluated low-rank setting, rather than as evidence that the weight is universally necessary.

When the affine trajectories are fully unconstrained, or when the basis spans the complete T-dimensional temporal space, $\omega[t]$ becomes algebraically redundant. Moreover, Eq. equation 25 exhibits a global scale ambiguity between $\boldsymbol{\omega}$ and the coefficient matrices. We therefore interpret $\omega[t]$ as a parameter-efficient temporal gate that complements the low-ra

C Projection and Diagnostic Protocols

For a TAB affine matrix $\mathbf{X} \in \mathbb{R}^{T \times C}$ and basis $\mathbf{L} \in \mathbb{R}^{T \times r_{\text{eff}}}$, define the orthogonal projector

$$\mathbf{P}_L = \mathbf{L}\mathbf{L}^\dagger, \qquad \widehat{\mathbf{X}} = \mathbf{P}_L\mathbf{X}. \tag{27}$$

The relative projection error and retained energy are

$$\epsilon(\mathbf{X}) = \frac{\|\mathbf{X} - \widehat{\mathbf{X}}\|_F}{\|\mathbf{X}\|_F}, \qquad E(\mathbf{X}) = 1 - \epsilon(\mathbf{X})^2. \tag{28}$$

For joint scale and bias, the unbalanced metric applies this definition to $[\mathbf{\Gamma} \mid \mathbf{B}]$. Because $\|\mathbf{\Gamma}\|_F$ can dominate $\|\mathbf{B}\|_F$, the balanced metric instead uses

$$\mathbf{X}_{\mathrm{bal}} = \left[\frac{\mathbf{\Gamma}}{\|\mathbf{\Gamma}\|_F} \;\middle|\; \frac{\mathbf{B}}{\|\mathbf{B}\|_F} \right], \tag{29}$$

with zero-norm blocks omitted. We compute energy separately for each normalization layer and report the unweighted mean and minimum across layers. The fixed-basis projection metrics use the uncentered affine matrices. Temporal centering, with $\mathbf{H} = \mathbf{I}_T - \mathbf{1}_T\mathbf{1}_T^\top/T$, is used for visualization and for the separate intrinsic-rank analysis described below.

For the variance diagnostic, direct concatenation of activations over steps $1{:}t$ is the reference. For every layer, cumulative step, batch, and channel, we compute

$$e_{\mathrm{rel}} = \frac{|\widehat{\sigma}^2_{1:t} - \sigma^2_{\mathrm{concat},1:t}|}{\sigma^2_{\mathrm{concat},1:t} + 10^{-12}}. \tag{30}$$

Figure 2 reports medians and 95th percentiles over the stated batch–channel population. The variance diagnostic is computed using the max checkpoint of the CIFAR-10 ResNet-19 model.

D Experimental Setup

D.1 Datasets and Data Splits

CIFAR-10 and CIFAR-100. CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009) are static-image classification benchmarks containing $60{,}000$ RGB images with a spatial resolution of 32×32. Both datasets use the official split of $50{,}000$ training images and $10{,}000$ test images. CIFAR-10 contains 10 classes, with $5{,}000$ training images and $1{,}000$ test images per class. CIFAR-100 contains 100 classes, with 500 training images and 100 test images per class. We use the official training sets for optimization and report classification accuracy on the corresponding official test sets.

DVS-CIFAR10. DVS-CIFAR10, also referred to as CIFAR10-DVS (Li et al., 2017), is a neuromorphic object-recognition dataset obtained by presenting CIFAR-10 images to a dynamic vision sensor. It contains $10{,}000$ event-stream samples from 10 object classes, with an original sensor resolution of 128×128. Following the evaluation protocol used by TAB (Jiang et al., 2024), we use $9{,}000$ samples for training and $1{,}000$ samples for testing. The event data are loaded through the SpikingJelly framework (Fang et al., 2023), and their spatial resolution is reduced from 128×128 to 48×48 before being provided to the network.

ImageNet. For large-scale evaluation, we use the ILSVRC-2012 ImageNet-1K classification benchmark (Deng et al., 2009), which contains $1{,}281{,}167$ training images and $50{,}000$ validation images from 1,000 object categories. The official training split is used for model optimization, and top-1 classification accuracy is reported on the official validation split. No images from the validation split are used to update the network parameters. During evaluation, each image is resized and center-cropped to 224×224 before being provided to the ResNet-34 model.

D.2 Data Preprocessing

For CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), training images are normalized and augmented using random cropping and random horizontal flipping. Test images are normalized without stochastic augmentation. Following the controlled comparison protocol used in the main experiments, we do not use Cutout (DeVries & Taylor, 2017). For DVS-CIFAR10, the event streams are converted into frame-based inputs over the specified simulation horizon and spatially resized to 48×48. For ImageNet, we apply the standard training-time image augmentation used for ResNet models, while evaluation uses a deterministic 224×224 center crop. The training and evaluation splits remain strictly disjoint for all datasets.(He et al., 2016; Fang et al., 2021a)

Datasets and architectures. We evaluate LRTB on the static-image benchmarks CIFAR-10, CIFAR-100, and ImageNet, as well as the neuromorphic DVS-CIFAR10 benchmark. Dataset descriptions, data splits, and preprocessing protocols are provided in Appendix D.1. We use VGG and

Table 3: Training hyperparameters for each dataset.

Dataset	Optimizer	Epochs	Scheduler	Learning Rate	Batch Size	Momentum	Weight Decay
CIFAR-10	SGD	200	CosineAnnealingLR(T=200)	0.02	64	0.9	0.0005
CIFAR-100	SGD	200	CosineAnnealingLR(T=200)	0.02	64	0.9	0.0005
DVS-CIFAR10	SGD	200	CosineAnnealingLR(T=200)	0.05	64	0.9	0.0005

ResNet architectures for the static datasets and a 7-layer CNN for DVS-CIFAR10, following the architectures specified in Table 1.

Training and evaluation. We employ discrete-time LIF neurons with leak factor $\lambda = 0.9$, firing threshold $V_{\mathrm{th}} = 1.0$, and hard reset to zero. The spike function is optimized using a triangular surrogate gradient with width parameter $\gamma = 1.0$. All reported CIFAR experiments use $T = 4$ simulation steps. The output layer is non-spiking, and predictions are obtained by averaging the per-step logits. CIFAR models are trained for 200 epochs using SGD with learning rate 0.02, momentum 0.9, weight decay 5×10^{-4}, batch size 64, and cosine-annealing learning-rate scheduling.ImageNet training uses eight NVIDIA A100 GPUs with 96 samples per GPU; batch statistics are synchronized across devices (Zheng et al., 2021). Evaluation uses a 224×224 center crop. The network is trained for 80 epochs using the AdamW optimizer with an initial learning rate of 0.0001 and a weight decay of 0.002.

Comparison methods. We compare LRTB with representative SNN normalization methods, including NeuNorm, BNTT (Kim & Panda, 2021), tdBN (Zheng et al., 2021), TEBN (Duan et al., 2022), and TAB (Jiang et al., 2024). Representative ANN-to-SNN conversion methods are also included for broader comparison. Results of competing methods are taken from their original publications. As training configurations may differ across studies, we emphasize comparisons using identical network architectures and simulation lengths, with TAB serving as the closest normalization baseline.

Normalization implementation. All per-step means and population variances are computed over the batch and spatial dimensions for each channel. Cumulative moments are updated in FP32 using Appendix A, and ε is added only when normalizing. ImageNet uses synchronized per-step moments across eight GPUs. Running moments, rather than test-batch moments, are used at inference. The basis is generated once for each simulation horizon, coefficient matrices are initialized as in Appendix B, and $r = 3$ unless otherwise stated.

D.3 Randomness and Reporting Protocol

Following the evaluation protocol commonly used by the closest SNN normalization baselines, the main tables report top-1 accuracy from training runs using fixed random seeds. The seed used for each experiment is specified in its released configuration file. We seed Python, NumPy, PyTorch, and all CUDA devices.

The closest temporal-normalization baselines, including BNTT, TEBN, tdBN, and TAB, report single-run classification accuracy without cross-seed error bars. We follow the same protocol to maintain direct comparability with their published results. Nevertheless, because the reported results do not characterize variation across independent training runs, small numerical differences should not be interpreted as formal evidence of statistical significance.

The evidence for LRTB is not based on a single dataset–architecture pair. Under matched architectures and simulation lengths, LRTB improves over the reported TAB results in all six evaluated settings, with improvements ranging from 0.42 to 1.37 percentage points. The controlled component ablation additionally keeps the architecture, data pipeline, optimizer, training schedule, and random seed fixed, while changing only the evaluated normalization component.

Table 4: Performance across different simulation horizons. The baseline and LRTB use the same architecture, training configuration, and simulation horizon. The target rank is $r = 3$, with $r_{\text{eff}} = \min(r, T)$. Δ denotes the accuracy improvement of LRTB over the corresponding same-horizon baseline.

Dataset	Architecture	T	r_{eff}	r_{eff}/T	Baseline (%)	LRTB (%)	Δ (pp)
CIFAR-10	ResNet-19	2	2	1.00	94.73	95.51	+0.78
CIFAR-10	ResNet-19	4	3	0.75	94.76	96.13	+1.37
CIFAR-10	ResNet-19	6	3	0.50	94.81	95.94	+1.13
CIFAR-100	ResNet-19	2	2	1.00	76.31	76.86	+0.55
CIFAR-100	ResNet-19	4	3	0.75	76.81	78.06	+1.25
CIFAR-100	ResNet-19	6	3	0.50	76.82	77.73	+0.91

E Implementation and Reproducibility Details

We train the models using SGD with cosine annealing of the learning rate on CIFAR-10, CIFAR-100, and DVS-CIFAR10 (Loshchilov & Hutter, 2016; Bottou, 2012); the dataset-specific settings are summarized in Table 3.

F Evaluation Across Different Simulation Horizons

To evaluate the behavior of LRTB under different simulation horizons, we additionally conduct experiments with $T = 2$ and $T = 6$, complementing the default $T = 4$ results reported in the main paper. All training settings, including the network architecture, data preprocessing, optimizer, learning-rate schedule, and number of training epochs, are kept unchanged unless otherwise specified. For every comparison, LRTB and the baseline use the same simulation horizon.

The target temporal rank is set to $r = 3$. Following the definition $r_{\text{eff}} = \min(r, T)$, the effective rank is therefore 2 when $T = 2$, while it remains 3 for $T = 4$ and $T = 6$. The corresponding rank-to-horizon ratios are 1.0, 0.75, and 0.5, respectively. Thus, although the absolute rank remains unchanged between $T = 4$ and $T = 6$, the $T = 6$ setting imposes a stronger relative low-rank constraint on the temporal affine trajectories.

As shown in Table 4, LRTB consistently outperforms the corresponding baseline at the same simulation horizon. The $T = 6$ accuracy is slightly lower than that obtained with $T = 4$. One plausible explanation is that fixing the rank to $r = 3$ while increasing the horizon from four to six steps produces a more restrictive temporal representation: the affine trajectories are represented using three basis components over six time steps, compared with three components over four time steps at $T = 4$.

We emphasize that the observed reduction from $T = 4$ to $T = 6$ should not be interpreted as conclusive evidence that the temporal rank is the sole cause of the difference, since other effects associated with a longer simulation horizon may also influence optimization. Instead, the result suggests a practical trade-off between temporal compression and classification accuracy. Increasing the rank for longer horizons may provide additional capacity, which we leave as an interesting direction for further investigation.

At $T = 2$, LRTB achieves 95.51 on CIFAR-10 and 76.86 on CIFAR-100, outperforming TAB by 0.78 and 0.55 percentage points, respectively. Because $r_{\text{eff}} = T = 2$, this setting does not impose dimensional compression along the temporal axis, although it still uses the structured exponential basis.

At $T = 6$, LRTB achieves 95.94 on CIFAR-10 and 77.73 on CIFAR-100. These results are slightly below the corresponding $T = 4$ LRTB accuracies, but remain higher than the same-horizon TAB baselines by 1.13 and 0.91 percentage points. Unlike the $T = 4$ setting, where $r/T = 0.75$, the $T = 6$ setting uses $r/T = 0.5$. The lower relative rank provides a more compact temporal representation but may restrict the capacity to model increasingly complex temporal affine trajectories over the longer horizon.