Intrinsically Stable Spiking Neural Networks: Overcoming the Performance Barrier in the Absence of Batch Normalization

Ruichen Ma[1], Xiaoyang Zhang[2], Jian Bai[2], Guanchao Qiao[1], Liwei Meng[1], Ning Ning[1], Yang Liu[1], and Shaogang Hu[1,3⋆]

[1] University of Electronic Science and Technology of China (UESTC)
[2] Beijing Institute of Remote-Sensing Equipment
[3] Shenzhen Institute for Advanced Study, UESTC
https://github.com/Ruichen0424/IS-SNN

Abstract. The performance of deep spiking neural networks (SNNs) often relies on batch normalization (BN). However, the advanced dynamic BN variants used in state-of-the-art models introduce runtime multiplications, which weaken the hardware-efficiency motivation of SNNs. To address this tension, we identify catastrophic firing-rate decay as a primary cause of severe performance degradation in normalization-free SNNs. Guided by this insight, this work proposes the Intrinsically Stable SNN (IS-SNN) architecture, which removes activation-normalization layers by enforcing signal homeostasis through topology-aware weight standardization and modified residual connections. By folding the standardization operations into static weights offline, IS-SNN removes the runtime statistics tracking and multiplications introduced by activation normalization, restoring an accumulation-oriented inference datapath. Comprehensive experiments show that IS-SNN achieves performance competitive with or superior to computationally expensive dynamic BN techniques across VGG, ResNet, and Transformer-based models. Notably, it achieves a competitive accuracy of 68.05% on ImageNet and overcomes the severe depth limitations of prior BN-free attempts. Together with a 96.4% reduction in FPGA lookup table resource consumption for neuron implementations, these results support IS-SNN as a practical framework for building accurate and hardware-friendly deep neuromorphic systems.

Keywords: Spiking neural networks · Batch-normalization-free architectures · Neuromorphic computing

1 Introduction

Spiking neural networks (SNNs), inspired by the information processing mechanisms of the brain, operate using asynchronous, event-driven spikes [26, 44, 46]. This paradigm offers the potential to realize a new generation of highly energy-efficient intelligent systems [34]. When implemented on neuromorphic hardware,

⋆ Corresponding author. (Email: sghu@uestc.edu.cn)

SNNs can achieve substantially lower power consumption than conventional artificial neural networks (ANNs) [5,11,18,28,32]. While early approaches successfully converted pre-trained ANNs to SNNs [14,15,20,40], they typically suffered from high inference latency. Consequently, direct training via surrogate gradients has emerged as a dominant methodology for low-latency inference [29,41], delivering highly competitive accuracy.

However, the success of state-of-the-art deep SNNs has become increasingly dependent on specialized batch normalization (BN) variants to stabilize training and improve performance [6,7,17,21,22,36,42,48]. This reliance creates a tension with the hardware-efficiency motivation of neuromorphic computing. While standard BN in SNNs can be fused into preceding synaptic weights or neuronal thresholds during inference [12,31], leading models often rely on advanced, batch- or time-dependent BN variants such as TEBN [7] and TAB [21]. These dynamic methods are difficult to fold into static parameters because their normalization statistics must be recomputed at runtime. Although advanced neuromorphic chips technically support multiplication operations, these incur substantially higher area and energy costs than simple adders [4,27]. By reintroducing runtime multiplications and statistical tracking at every timestep, dynamic BN weakens the accumulation-oriented datapath that motivates SNN deployment.

This dependency creates an important dilemma: simply removing activation normalization is not a viable solution. As corroborated by forward signal analysis, deep SNNs lacking statistics control face a signal-propagation barrier: catastrophic firing-rate decay. Without normalization, pre-activation variances can drift across layers, causing neuronal firing rates to either vanish or saturate. This disrupts signal propagation and can lead to severe performance degradation or training collapse. Consequently, the field faces an undesirable trade-off between using costly, non-fusible dynamic normalization and accepting severe limitations on network depth. Prior attempts at BN-free SNNs have been largely restricted to shallow models and simpler datasets and have not yet established an effective solution for modern deep architectures [31].

To address this conflict, we shift the perspective from optimization instability to signal survival. Motivated by the observation that catastrophic firing-rate decay is a primary bottleneck in normalization-free SNNs, this paper introduces the Intrinsically Stable SNN (IS-SNN). IS-SNN removes activation-normalization layers while maintaining competitive accuracy through intrinsic signal stabilization. Building on weight reparameterization and residual scaling, IS-SNN adapts these ideas to spiking dynamics through topology-aware weight standardization and modified residual connections that enforce signal homeostasis. By decoupling the training and inference phases, IS-SNN enables the standardization operations to be folded into static weights offline before deployment. The main contributions of this work are as follows:

- The proposed IS-SNN maintains signal homeostasis with topology-aware weight standardization and a modified residual connection. By decoupling training and inference, the standardization operations can be folded into static weights offline, introducing no inference-time normalization overhead.

- Comprehensive experiments demonstrate that IS-SNN achieves performance competitive with or superior to advanced, computationally expensive dynamic BN variants across diverse architectures, including VGG, ResNet, and Transformer-based models. Notably, it scales to deep networks such as ResNet-152 and achieves 68.05% accuracy on ImageNet, substantially improving over naive BN-free baselines.
- A dedicated hardware analysis shows a 15% increase in training throughput and a 96.4% reduction in FPGA lookup table (LUT) resource consumption for neuron implementations, supporting the practical efficiency benefits of removing runtime normalization operations.

2 Related Work

Normalization in SNNs and Hardware Fusibility. Batch normalization (BN) [19] has been central to training high-performance neural networks. In SNNs, early adaptations included standard BN [9] and temporal extensions [37]. Standard static BN can be mathematically folded into synaptic weights or neuronal thresholds during inference, thereby preserving the inference efficiency of SNNs [12,31]. However, to close the accuracy gap with ANNs, modern deep SNNs often rely on advanced time- or batch-dependent BN variants, such as tdBN [48], BNTT [22], TEBN [7], and TAB [21]. Because these methods compute normalization statistics dynamically at runtime, they are difficult to reduce to fixed offline parameters. This reintroduces runtime multiplications and weakens the accumulation-oriented hardware efficiency that motivates SNN deployment.

Activation-Normalization-Free SNNs. The overhead of dynamic BN has motivated research into BN-free SNNs. OTTT [43] uses Weight Standardization (WS), following normalization-free ANN designs, in the context of online training through time. Its primary goal is to reduce online training complexity, whereas our work addresses the different problem of stabilizing deep SNNs after removing activation normalization. Rather than treating WS as a generic weight reparameterization, IS-SNN derives topology-aware scaling factors from SNN signal propagation, neuron output variance, and residual variance growth under spiking dynamics. Other BN-free SNN attempts [31] have mainly been evaluated on shallow VGG architectures and small-scale datasets. Establishing a general framework for high-performance, activation-normalization-free SNNs that scales to modern deep architectures therefore remains an important open challenge.

Weight Reparameterization. Weight reparameterization provides another route to improving training stability without activation normalization. In conventional ANNs, Weight Standardization [33] was first introduced to smooth the loss landscape and accelerate optimization, particularly for micro-batch training. Normalizer-free ResNet studies further developed signal propagation analysis, scaled weight standardization, and residual branch scaling for deep ANNs without BN [2,3]. These works provide important foundations for normalization-free

network design. However, directly transferring these principles to SNNs is non-trivial because spike activations are discrete, temporally integrated, and governed by neuron-specific firing dynamics. In IS-SNN, WS is therefore adapted to maintain firing-rate homeostasis. The scaling coefficient γ_ℓ is derived from SNN-specific variance propagation, including the empirically estimated neuron output variance σ_g and topology-dependent residual accumulation. As confirmed by our ablations, naive WS is insufficient for deep SNNs, whereas this topology-aware formulation effectively counteracts firing-rate decay while allowing the standardization operations to be folded into static weights before deployment.

3 Method

3.1 Preliminaries of Spiking Neuron Dynamics

The core computational unit of an SNN is the spiking neuron (SN). Its dynamics encompass three stages: integration, firing, and reset:

$$H[t] = f(V[t-1], X[t]) \tag{1}$$

$$S[t] = \Theta(H[t] - V_{th}) \tag{2}$$

$$V[t] = H[t](1 - S[t]) + V_{reset}S[t] \tag{3}$$

where $X[t]$ is the input current at timestep t. $H[t]$ and $V[t]$ denote the membrane potentials immediately before and after spike generation, respectively. The output spike $S[t]$ is a binary event triggered when $H[t]$ exceeds the firing threshold V_{th}. The Heaviside step function is defined as $\Theta(x) = 1$ for $x > 0$ and $\Theta(x) = 0$ for $x \leq 0$. Following a spike, the membrane potential resets to V_{reset}.

The integration function $f(\cdot)$ in Eq. (1) defines the specific neuronal behavior. The standard Leaky Integrate-and-Fire (LIF) neuron is formulated as:

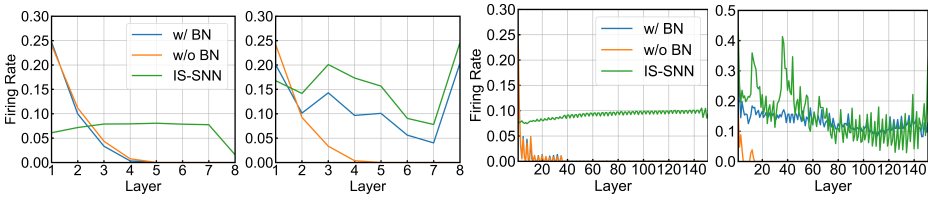
$$H[t]_{LIF} = V[t-1] - \frac{1}{\tau}(V[t-1] - V_{reset}) + X[t] \tag{4}$$

where τ is the membrane time constant. For neuromorphic datasets, the Parametric LIF (PLIF) model [10] was employed, where τ is learnable. The LIF model with decaying input similarly applies a $1/\tau$ coefficient to $X[t]$.

3.2 Forward Signal Analysis of Firing Rate

To investigate the failure mechanisms of deep SNNs trained without activation normalization, forward signal analysis was performed to track activation statistics during propagation. This analysis is important for networks that lack active mechanisms to regulate internal data distributions. In SNNs, the mean activation μ corresponds directly to the average firing rate, physically representing the network's overall activity level.

A deep neural network can be conceptualized as a sequence of stacked minimum repetition units (MRUs). In plain networks like VGG, an MRU consists



(a) Before training. **(b)** After training. **(c)** Before training. **(d)** After training.

Fig. 1: Layer-wise firing rates in SNNs. (a-b) VGG-9 on CIFAR-10 before and after training. Without BN, the network suffers from rapid firing-rate decay and fails to recover. Both the BN baseline and the proposed IS-SNN prevent this collapse. (c-d) SEW-ResNet-152 on CIFAR-100. A similar signal-decay phenomenon is observed in the deep architecture, where IS-SNN maintains stable signal propagation without BN.

of a Conv-BN-SN sequence. In residual networks, it comprises several residual blocks and a transition block. For information to propagate effectively, the firing rate should remain stable across successive MRUs. If the firing-rate ratio between adjacent MRUs, $c = \mu_{L+1}/\mu_L$, deviates substantially from 1, the rate will either vanish or saturate as the layer depth L increases: $\lim_{L\to+\infty} \mu_L = 0$ (when $0 \leq c < 1$) or $\lim_{L\to+\infty} \mu_L = 1$ (when $c > 1$). Both outcomes are detrimental. A vanishing rate indicates a broken signal path, while a saturated rate limits the network's expressive capacity. From an information-theoretic perspective, the entropy of the spike train, $H(S_L) = -\mu_L \log \mu_L - (1-\mu_L)\log(1-\mu_L)$, is maximized when $\mu_L = 0.5$ and approaches zero as μ_L approaches either 0 or 1.

This failure mode is empirically supported by Fig. 1. The figures plot the layer-wise firing rates for VGG-9 on CIFAR-10 and SEW-ResNet-152 on CIFAR-100, comparing models with BN, without BN, and with the proposed IS-SNN. Without BN, the naive baseline suffers from firing-rate decay across layers and fails to recover. In contrast, BN-equipped networks dynamically restore rates to a stable range during training. These results indicate that catastrophic firing-rate decay is a primary cause of signal collapse in deep SNNs lacking activation normalization. The key role of BN is to provide a mechanism for dynamic stabilization. This is analogous to homeostatic plasticity in biological systems, where neurons dynamically regulate their properties to maintain firing rates within a stable range [38]. This parallel suggests that maintaining a stable firing rate is also important for efficient information transmission in artificial spiking systems.

3.3 The IS-SNN Architecture

Motivated by the preceding analysis, the IS-SNN architecture is designed to stabilize the firing rate by controlling the pre-activation statistics layer by layer. Inspired by synaptic scaling, a biological homeostatic process that adjusts synaptic strengths to stabilize neuronal activity [39], IS-SNN employs Weight Standardization (WS) to remove data-dependent activation-normalization layers. This formulation decouples the training and inference phases, enabling deployment without inference-time normalization overhead.

Weight Standardization and Offline Reparameterization. To mimic the stabilizing effect of BN, IS-SNN aims to keep the pre-activation scale close to a standard-normal reference, $x_{pa} \sim \mathcal{N}(0,1)$. The mean μ_{pa} and variance σ^2_{pa} of the pre-activation are governed by the statistics of input spikes and weights:

$$\mu_{pa} = N\mu_{in}\mu_{W_i} \tag{5}$$

$$\sigma^2_{pa} = N\sigma^2_{in}(\sigma^2_{W_i} + \mu^2_{W_i}) \tag{6}$$

where N is the fan-in. To approach the target scale $x_{pa} \sim \mathcal{N}(0,1)$, the weights are regulated to maintain a zero mean, $\mu_{W_i} = 0$, and a scaled variance, $\sigma^2_{W_i} = 1/(N\sigma^2_{in})$. We enforce these conditions using weight standardization:

$$\hat{W}_{i,j} = \gamma_\ell \cdot \frac{W_{i,j} - \mu_i}{\sqrt{N\sigma_i^2 + \epsilon}} \tag{7}$$

where $W_{i,j}$ are the original weights, μ_i and σ_i^2 are their statistical mean and variance, $\hat{W}_{i,j}$ are the standardized weights, and $\epsilon = 10^{-4}$ ensures numerical stability. The critical component of this formulation is the scaling factor γ_ℓ, which must be set appropriately for each layer ℓ to control the pre-activation variance. A convolutional layer incorporating this technique is termed WS-Conv.

During training, the learnable weights $W_{i,j}$ continuously update, causing their statistics μ_i and σ_i^2 to fluctuate. Therefore, at the beginning of each training step, the original weights are dynamically standardized using Eq. (7). The network then uses the standardized weights $\hat{W}_{i,j}$ for forward propagation. Once training concludes, all parameters, including the weights, their statistics, and the scaling factor γ_ℓ, become fixed. This enables the standardization operations to be folded into static weights offline before deployment. During inference, the hardware uses these pre-scaled static weights $\hat{W}_{i,j}$. By decoupling normalization from the inference phase, the deployed IS-SNN is structurally identical to a raw BN-free network. It requires no runtime normalization statistics tracking or normalization-related multiplications, preserving the accumulation-oriented datapath of SNNs.

Topology-Aware Variance Propagation. With the WS mechanism established, its effectiveness depends on determining the scaling factor γ_ℓ for each layer across different network topologies. For clarity in derivation, we introduce three core parameters governing variance propagation:

- σ_ℓ: The theoretically estimated input standard deviation for layer ℓ. It is a topology-dependent constant and remains invariant across timesteps.
- γ_ℓ: The layer-wise scaling factor, defined as $\gamma_\ell \equiv 1/\sigma_\ell$, which ensures the weights satisfy the variance condition $\sigma^2_{W_i} = 1/(N\sigma^2_\ell)$ required by Eq. (7).
- σ_g: A fixed constant representing the empirically estimated output standard deviation of a specific spiking neuron type under a standard-normal input. It is an intrinsic neuronal property, invariant across layers and timesteps.

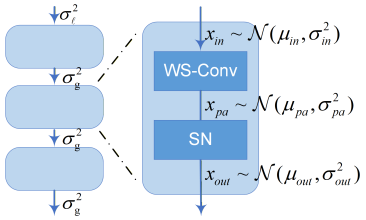


Fig. 2: The minimum repetition unit (MRU) of the plain IS-SNN, which replaces the Conv-BN-SN block with a weight-standardized convolutional layer followed by a spiking neuron.

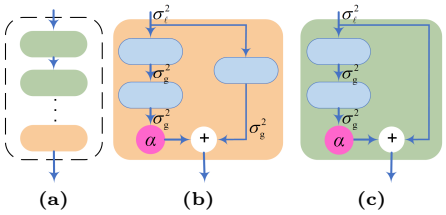


Fig. 3: The (a) MRU of the residual IS-SNN consists of one (b) transition block and several (c) residual blocks. The modified residual connection incorporates a scaling factor α to control variance growth.

The calculation of σ_ℓ depends on the network architecture. In plain non-residual networks such as VGG, the output of layer ℓ serves directly as the input to layer $\ell+1$, as shown in Fig. 2. Consequently, the variance remains constant across layers: $\sigma^2_{in,\ell+1} = \sigma^2_{out,\ell} = \sigma^2_g$. The scaling factor for any layer ℓ is therefore set to $\gamma_\ell \equiv 1/\sigma_g$. Conversely, residual networks compound variance through addition operations. This can cause signal variance to grow with successive blocks and destabilize BN-free training. To control this growth, we introduce a modified residual connection, as shown in Fig. 3:

$$x_{\ell+1} = \alpha \cdot \mathrm{SN}(f(x_\ell)) + x_\ell \tag{8}$$

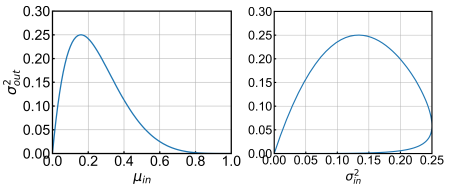
where α is a scaling coefficient that regulates the residual branch's contribution (uniformly set to 0.5 across all architectures, as justified in Sec. 4.3). Within a continuous sequence of residual blocks, the tracked input variance follows $\sigma^2_{\ell+1} = \alpha^2 \cdot \sigma^2_g + \sigma^2_\ell$. At the beginning of a stage, the transition block resets the statistics to $\sigma^2_{reset} = (1+\alpha^2) \cdot \sigma^2_g$. By analytically tracking the propagated variance σ^2_ℓ layer by layer, the scaling factor $\gamma_\ell = 1/\sigma_\ell$ can be computed throughout the ResNet topology.

Empirical Estimation of σ_g^2. The design of IS-SNN depends on the neuron's base output variance σ^2_g. Unlike ReLU units, which have a transparent linear relationship between input and output variance [2], the dynamics of spiking neurons are highly non-linear and timestep-dependent, making analytical derivation difficult. Therefore, σ^2_g is approximated empirically by stimulating various neuron models with random pre-activations drawn from $\mathcal{N}(0,1)$. Table 1 lists the time-averaged output variance $\overline{\sigma^2_g}$ for different LIF neuron configurations. This simulation-based method can be applied to estimate the corresponding σ^2_g for other neuron models. See the Appendix for more details.

Impact of Auxiliary Network Components. Signal propagation is also affected by other network components, such as max-pooling layers. The relationship between input and output variance is particularly complex for binary spike

Table 1: Empirically estimated time-averaged output variance, $\overline{\sigma_g^2}$, for different Leaky Integrate-and-Fire (LIF) models.

Model	Decay Input	$\overline{\sigma_g^2}$		
		T=4	T=8	T=16
LIF	×	0.1234	0.1195	0.1174
($\tau = 2$)	√	0.0290	0.0298	0.0302



(a) $\sigma^2_{out} - \mu_{in}$ **(b)** $\sigma^2_{out} - \sigma^2_{in}$

Fig. 4: Simulated output variance σ^2_{out} of a max-pooling layer processing binary spike inputs.

activations. As the input firing rate μ_{in} increases from 0 to 1, the input variance σ^2_{in} first increases from 0 to a maximum of 0.25 before decreasing back to 0. This complex, non-monotonic relationship is detailed in Fig. 4. While max-pooling is supported in IS-SNN, caution is recommended due to this complex interaction. Downsampling is more reliably achieved using strided convolutions. For the first encoding layer, the scaling factor γ_1 is uniformly set to 1 for simplicity and generalizability. By analyzing the effect of each component, the theoretically expected input variance σ^2_ℓ for each layer can be estimated, completing the specification of the IS-SNN architecture.

4 Experiments

4.1 Experimental Setup

Experiments were conducted on the static image classification datasets CIFAR [23] and ImageNet [35], as well as the neuromorphic datasets DVS-Gesture [1] and CIFAR10-DVS [24]. All models were implemented in SpikingJelly [8] and trained with surrogate gradients, using automatic mixed precision for acceleration. For the CIFAR datasets, models were trained for 256 epochs with an initial learning rate of 0.02 and a weight decay of 5e-4. For ImageNet, an initial learning rate of 0.1 and a weight decay of 2e-5 were used. Results for both 128- and 400-epoch schedules with standard augmentation techniques [16] were reported to evaluate the architecture under different training budgets. For static datasets, the standard LIF model was used. For DVS-Gesture, the setup from 7B-Net [9] was followed. The spike-element-wise (SEW) ResNet structure [9] was adopted for residual architectures. Comprehensive hyperparameter configurations are provided in the Appendix.

4.2 Performance Evaluation

As predicted by the forward signal analysis, the IS-SNN architecture maintains stable firing rates across layers both at initialization and after training, as shown in Fig. 1. This intrinsic signal stability translates into strong performance across deep architectures, as demonstrated in Tabs. 2 and 3.

Table 2: Performance comparison of IS-SNN against state-of-the-art normalization methods and a baseline without normalization (w/o BN). The w/o BN results show severe performance degradation on deep or complex models. IS-SNN achieves accuracy competitive with or superior to advanced, non-fusible normalization methods while introducing no inference-time normalization overhead.

Dataset	Model	Method	Timesteps	Accuracy (%)
CIFAR-10	Spike-Norm [36]	VGG-16	2500	91.55
	tdBN [48]	ResNet-19	6 / 4 / 2	93.16 / 92.92 / 92.34
	BNTT [22]	VGG-9	25 / 20	90.5 / 90.3
	TET [6]	ResNet-19	6 / 4 / 2	94.50 / 94.44 / 94.16
	w/o BN	VGG-9	4	10.00
		ResNet-19	6 / 4 / 2	91.98 / 90.46 / 86.87
	IS-SNN	VGG-9	4	**92.91**
		ResNet-19	6 / 4 / 2	**94.51 / 94.32 / 93.96**
CIFAR-100	Spike-Norm [36]	VGG-16	2500	70.90
	tdBN [6]	ResNet-19	6 / 4 / 2	71.12 / 70.86 / 69.41
	TET [6]	ResNet-19	6 / 4 / 2	74.72 / 74.47 / 72.87
	TEBN [7]	ResNet-19	6 / 4 / 2	76.41 / 76.13 / 75.86
	w/o BN	ResNet-19	6 / 4 / 2	69.98 / 67.99 / 63.61
		ResNet-152	4	17.10
	IS-SNN	ResNet-19	6 / 4 / 2	**76.47 / 76.02 / 74.83**
		ResNet-152	4	**76.98**
DVS-Gesture	w/ BN [9]	7B-Net	16	97.92
	BN-free [31]	3 Conv	150	95.49
	w/o BN	7B-Net	16	77.08
	IS-SNN	7B-Net	16	**96.88**
CIFAR10-DVS	w/ BN	VGGSNN	8	78.1
	w/o BN	VGGSNN	8	10.0
	IS-SNN	VGGSNN	8	**77.7**
ImageNet	tdBN [48]	ResNet-34	6	63.72
	Spike-Norm [36]	ResNet-34	2500	65.47
	TET [6]	ResNet-34	4	68.00
	TEBN [7]	ResNet-34	4	68.28
	w/o BN	ResNet-34	4	32.58
	IS-SNN	ResNet-34	4	**66.61**
	IS-SNN (E400)	ResNet-34	4	**68.05**

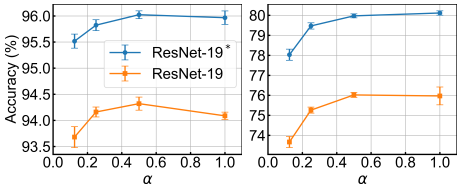
Table 2 presents a systematic comparison of IS-SNN against various normalization methods. The naive BN-free (w/o BN) models suffer severe degradation on deeper architectures: ResNet-152 on CIFAR-100 and ResNet-34 on ImageNet achieve accuracies of only 17.10% and 32.58%, respectively. In contrast, IS-SNN avoids this training failure and achieves performance highly competitive with advanced, hardware-intensive dynamic BN techniques. For ResNet-19, IS-SNN obtains substantial accuracy gains over the w/o BN baseline, with absolute improvements of 4.49% on CIFAR-10 and 8.58% on CIFAR-100, outperforming methods such as Spike-Norm, tdBN, and TET. On ImageNet, the 128-epoch schedule yields 66.61% accuracy. Extending the training schedule to 400 epochs

Table 3: Performance comparison with strong data augmentation, denoted by *. With a stronger training regime, IS-SNN achieves state-of-the-art results among the compared methods, including advanced non-fusible BN variants. Its application to Spikformer further demonstrates the applicability beyond convolutional networks.

Dataset	Model	Architecture	Timesteps	Accuracy (%)
CIFAR-10	TC [49]	VGG-16	-	92.68
	RMP [13]	ResNet-20	2048	91.36
	DSpike [25]	ResNet-18*	6 / 4 / 2	94.25 / 93.66 / 93.13
	Spikformer [50]	4-384*	4	95.19
	TEBN [7]	VGG-11	4	93.96
		ResNet-19*	6 / 4 / 2	95.60 / 95.58 / 95.45
	TAB [21]	VGG-11	4	94.73
		ResNet-19*	6 / 4 / 2	96.09 / 95.94 / 95.62
	w/o BN	VGG-11*	4	10.00
		ResNet-19*	6 / 4 / 2	92.73 / 89.94 / 85.00
	IS-SNN	VGG-11*	4	**95.06**
		ResNet-19*	6 / 4 / 2	**96.12** / **96.02** / **95.65**
		4-384*	4	**95.43**
CIFAR-100	BNTT [22]	VGG-11	50 / 30	66.6 / 65.8
	RMP [13]	ResNet-20	2048	67.82
	DSpike [25]	ResNet-18*	6 / 4 / 2	74.24 / 73.35 / 71.68
	Spikformer [50]	4-384*	4	77.86
	TEBN [7]	VGG-11	4	74.37
		ResNet-19*	6 / 4 / 2	78.76 / 78.71 / 78.07
	TAB [21]	VGG-11	4	75.89
		ResNet-19*	6 / 4 / 2	76.82 / 76.81 / 76.31
	w/o BN	VGG-11*	4	1.00
		ResNet-19*	6 / 4 / 2	74.18 / 70.60 / 65.58
	IS-SNN	VGG-11*	4	**77.13**
		ResNet-19*	6 / 4 / 2	**80.72** / **79.97** / **79.03**
		4-384*	4	**78.92**

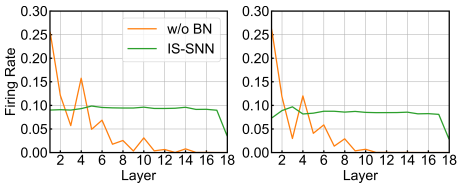
further improves the accuracy to 68.05%, surpassing TET (68.00%) and approaching the computationally expensive TEBN (68.28%). Considering that IS-SNN removes runtime normalization multiplications during deployment, these results indicate a favorable performance-efficiency trade-off for deep SNNs. On the neuromorphic DVS-Gesture dataset, IS-SNN also substantially outperforms the w/o BN baseline and surpasses prior BN-free attempts. To isolate the effect of learnable temporal dynamics, PLIF neurons were further replaced with standard LIF neurons under the same setting. IS-SNN still achieves 95.14%, remaining competitive with the corresponding 95.83% w/ BN baseline. IS-SNN was also evaluated on CIFAR10-DVS using the VGGSNN architecture, where IS-SNN achieves 77.7%, while the standard VGGSNN without BN fails to train.

To probe the potential of IS-SNN under stronger training regimes, further experiments were conducted on the CIFAR datasets using Mixup [47] and Cutmix [45], as detailed in Tab. 3. With ResNet-19, IS-SNN achieves state-of-the-art accuracies of 96.12% on CIFAR-10 and 80.72% on CIFAR-100 among the



(a) CIFAR-10. (b) CIFAR-100.

Fig. 5: Ablation study on the scaling factor α. The value $\alpha = 0.5$ consistently yields high accuracy with low variance across 48 total runs. The small gap between $\alpha = 0.5$ and $\alpha = 1.0$ indicates that the method is not highly sensitive to precise α tuning.



(a) ResNet-19A. (b) ResNet-19B.

Fig. 6: Initial layer-wise firing rates for two modified SEW-ResNet-19 variants on CIFAR-100. As the number of transition blocks increases, the baseline exhibits stronger rate decay, while IS-SNN remains stable across these structural variants.

compared methods, outperforming recent advanced methods such as TAB and TEBN. Additionally, IS-SNN can be applied to Spikformer, a Transformer-based architecture, to improve performance. As shown in Tab. 3, IS-SNN stabilizes the attention-based topology and improves performance on both CIFAR datasets, demonstrating its applicability beyond convolutional structures.

4.3 Ablation Study

Impact of the residual scaling factor α. The proposed residual connection is defined as $x_{\ell+1} = \alpha \cdot \mathrm{SN}(f(x_\ell)) + x_\ell$, where α controls the variance growth rate across residual blocks. To determine a suitable value, experiments were conducted on CIFAR datasets with $\alpha \in \{0.125, 0.25, 0.5, 1.0\}$. These values were chosen as integer powers of 2, so the scaling operation can be implemented by an efficient bit-shift rather than standard multiplication. To obtain statistically robust conclusions, each parameter setting was tested across three different seeds, yielding 48 full training runs in total. As shown in Fig. 5, setting $\alpha = 0.5$ consistently achieves strong performance with low variance. The performance gap between $\alpha = 0.5$ and $\alpha = 1.0$ is small ($< 0.2\%$), indicating a broad solution plateau. This robustness allows us to adopt a unified $\alpha = 0.5$ across different network backbones without exhaustive, architecture-specific hyperparameter sweeps.

Robustness to network structure. To verify whether additional transition blocks exacerbate firing-rate decay, ResNet-19 was modified to create two variants. While all three networks share 19 layers, the standard ResNet-19 (denoted as 3C128-3C256-2C512) contains two transition blocks. In contrast, the specially designed variants ResNet-19A (1C32-1C64-2C128-2C256-2C512) and ResNet-19B (1C32-1C64-1C96-1C128-2C256-2C512) contain four and five transition blocks, respectively. As shown in Tab. 4 and Fig. 6, networks lacking activation normalization degrade rapidly as the number of transition blocks increases, with ResNet-19B failing to converge (1.00%). These findings indicate that BN-free

Table 4: Accuracy comparison for the ablation study on the number of transition blocks on CIFAR-100. The w/o BN baseline degrades as the number of transition blocks increases, while IS-SNN maintains high performance across the tested variants.

Architecture	# Transition Blocks	# Params	Acc (%)	
			w/o BN	IS-SNN
ResNet-19*	2	14.7 M	70.60	**79.97**
ResNet-19A*	4	13.2 M	68.75	**79.02**
ResNet-19B*	5	13.1 M	1.00	**78.71**

training becomes increasingly unstable as signal decay accumulates across repeated structural transitions, consistent with the behavior observed in Fig. 1. In contrast, IS-SNN maintains high accuracy and stable firing rates across all topological variants, supporting its robustness to structural changes.

Naive Weight Standardization vs. IS-SNN. A key component of IS-SNN is the topology-aware derivation of the scaling coefficient γ_ℓ. To validate its necessity, an ablation study was conducted where γ_ℓ was fixed to 1 for all layers. This setting reduces the method to naive WS, which normalizes weight magnitude without explicitly addressing signal decay. The results in Tab. 5 show that naive WS is insufficient for deep SNNs. It leads to severe performance degradation for ResNet and complete training collapse for VGG. This confirms that calculating γ_ℓ based on topology-aware variance propagation is important for BN-free deep architectures. The sensitivity of IS-SNN to possible mis-estimation of the neuron variance parameter σ_g was further evaluated. Using ResNet-19 on CIFAR-10/100, random perturbations were injected into σ_g to induce deviations of up to 10% for each layer during training. The resulting accuracy drops were only 0.23% and 0.31%, respectively, indicating that IS-SNN remains robust to deviations from the estimated σ_g.

Synergy with Zero-Overhead Regularization. By eliminating dynamic BN, IS-SNN also removes the stochastic regularization effects provided by batch-wise statistics, a known trade-off for zero inference-time normalization cost. However, because IS-SNN provides a stable baseline, it can be combined with pure training-phase stochastic regularization. To validate this, standard Dropout was applied to ResNet-19 on CIFAR-10/100, yielding consistent accuracy gains of $+0.12\%$ and $+0.21\%$, respectively. Since Dropout is deactivated during inference, these improvements introduce no additional inference overhead. This suggests that IS-SNN is compatible with lightweight regularization techniques, providing a way to improve accuracy without compromising inference efficiency.

4.4 Hardware and Efficiency Benefits

Training efficiency. Activation normalization introduces additional computational overhead during training, whether executed on server-side GPUs or through

Table 5: Ablation on the scaling coefficient γ_ℓ. Forcing $\gamma_\ell \equiv 1$ reduces the model to naive Weight Standardization, leading to severe degradation or complete collapse. This confirms the importance of topology-aware variance scaling for SNNs.

Dataset	Structure	Acc (%)	
		$\gamma \equiv 1$	IS-SNN
CIFAR10	VGG-11*	10.00	**95.06**
	ResNet-19	92.05	**94.32**
CIFAR100	VGG-11*	1.00	**77.13**
	ResNet-19	70.69	**76.02**

on-chip online learning [43]. Experiments were conducted on GPUs to quantify this cost. For ResNet-34 on ImageNet, the network with standard BN achieved a training throughput of 401 imgs/sec/GPU. In contrast, IS-SNN achieved 461 imgs/sec/GPU, corresponding to a 15% increase in training speed and a 17% reduction in memory consumption. Because advanced dynamic BN variants introduce higher overhead than standard BN, these results suggest that IS-SNN can improve training efficiency while maintaining competitive accuracy.

Inference efficiency. To evaluate inference overhead on resource-constrained hardware, deployment was simulated on an Xilinx Virtex-7 FPGA. We compared two LIF kernels: (1) a BN-variant-enabled neuron, which requires both a multiplier and an adder for the dynamic calculations introduced by non-fusible BN variants, and (2) the proposed IS-SNN neuron. Because IS-SNN folds the standardization operations into synaptic weights offline during compilation, its neuron kernel requires only an adder for membrane potential accumulation and a comparator for thresholding. It therefore removes the runtime multiplication operators introduced by activation normalization. Using standard time-division multiplexing technology [30] to compile and deploy these kernels, the IS-SNN implementation achieved a 96.4% reduction in LUT resource consumption. This comparison isolates the local datapath operations and does not include the additional overhead that dynamic BN variants may incur for storing and updating temporal statistics; the actual savings may therefore be greater. Further hardware analysis and mathematical cost breakdowns are provided in the Appendix.

5 Limitations and Discussion

The IS-SNN architecture removes inference-time normalization overhead through offline reparameterization, which assumes that the weights remain fixed after training. Thus, IS-SNN is particularly suitable for inference-focused edge deployment, whereas online learning may require additional hardware support for efficient statistics tracking. The hardware analysis in this work focuses on the arithmetic and resource costs introduced by activation normalization, especially the runtime multiplications required by dynamic BN variants. Although the reported FPGA results show substantial LUT savings for neuron implementations,

complete end-to-end energy evaluation would also need to account for memory access, routing, and system-level scheduling. Such full-system evaluation on dedicated neuromorphic hardware remains an important direction for future work.

Nevertheless, by removing computationally expensive normalization operations during inference, IS-SNN helps reduce the tension between high performance and hardware efficiency in deep SNNs. This makes accurate SNN deployment on resource-constrained neuromorphic hardware more practical, especially for power-constrained edge applications where latency and energy are primary bottlenecks, such as autonomous robotics and wearable sensors. The connection between IS-SNN and homeostatic regulation also suggests that neuroscience-inspired stability principles can provide useful guidance for efficient systems.

This work opens several promising avenues for future research. First, the stable signal-propagation behavior provided by IS-SNN can serve as a baseline for investigating trade-offs among firing rate, spike sparsity, and task performance in SNNs. Second, future research could adapt these intrinsic stabilization mechanisms to hardware-friendly online learning paradigms. Third, extending IS-SNN beyond vision benchmarks to sequence-centric tasks and to unsupervised or self-supervised learning may further test the generality of BN-free SNNs.

6 Conclusion

This work addresses the tension between performance and hardware efficiency in deep spiking neural networks, which arises from the reliance on computationally expensive, non-fusible BN variants. Building on the observation that catastrophic firing-rate decay is a primary cause of signal collapse in networks lacking normalization, the proposed IS-SNN architecture removes activation-normalization layers by enforcing intrinsic signal stability through topology-aware weight standardization and offline reparameterization. Comprehensive experiments show that IS-SNN achieves performance competitive with or superior to advanced dynamic BN techniques across diverse network topologies, including VGG, ResNet, and Transformers, on benchmarks ranging from CIFAR to ImageNet. Its effectiveness in deep architectures is further supported by a 35.47% accuracy improvement on ImageNet over a naive BN-free baseline. By delivering highly competitive accuracy together with no inference-time normalization overhead and a 96.4% reduction in FPGA LUT resource consumption for neuron implementations, this work provides a practical framework for designing SNNs that are accurate, hardware-friendly, and biologically motivated.

Acknowledgements

This work was supported in part by the Brain Science and Brain-like Intelligence Technology-National Science and Technology Major Project 2022ZD0209700 and in part by Sichuan Provincial Industrial Development Fund under Grant 2025JB0 4 and in part by Shenzhen Natural Science Foundation under Grant JCYJ202506 04180428038.

References

1. Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., et al.: A low power, fully event-based gesture recognition system. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7243–7252 (2017)
2. Brock, A., De, S., Smith, S.L.: Characterizing signal propagation to close the performance gap in unnormalized resnets. In: International Conference on Learning Representations (2021)
3. Brock, A., De, S., Smith, S.L., Simonyan, K.: High-performance large-scale image recognition without normalization. In: International Conference on Machine Learning. pp. 1059–1071. PMLR (2021)
4. Chen, T., Zhang, Z., Ouyang, X., Liu, Z., Shen, Z., Wang, Z.: " bnn-bn=?": Training binary neural networks without batch normalization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 4619–4629 (2021)
5. Davies, M., Srinivasa, N., Lin, T.H., Chinya, G., Cao, Y., Choday, S.H., Dimou, G., Joshi, P., Imam, N., Jain, S., et al.: Loihi: A neuromorphic manycore processor with on-chip learning. Ieee Micro **38**(1), 82–99 (2018)
6. Deng, S., Li, Y., Zhang, S., Gu, S.: Temporal efficient training of spiking neural network via gradient re-weighting. In: International Conference on Learning Representations (2022)
7. Duan, C., Ding, J., Chen, S., Yu, Z., Huang, T.: Temporal effective batch normalization in spiking neural networks. Advances in Neural Information Processing Systems **35**, 34377–34390 (2022)
8. Fang, W., Chen, Y., Ding, J., Yu, Z., Masquelier, T., Chen, D., Huang, L., Zhou, H., Li, G., Tian, Y.: Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. Science Advances **9**(40), eadi1480 (2023)
9. Fang, W., Yu, Z., Chen, Y., Huang, T., Masquelier, T., Tian, Y.: Deep residual learning in spiking neural networks. Advances in Neural Information Processing Systems **34**, 21056–21069 (2021)
10. Fang, W., Yu, Z., Chen, Y., Masquelier, T., Huang, T., Tian, Y.: 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 (2021)
11. Furber, S.B., Galluppi, F., Temple, S., Plana, L.A.: The spinnaker project. Proceedings of the IEEE **102**(5), 652–665 (2014)
12. Guo, Y., Zhang, Y., Chen, Y., Peng, W., Liu, X., Zhang, L., Huang, X., Ma, Z.: Membrane potential batch normalization for spiking neural networks. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19420–19430 (2023)
13. Han, B., Srinivasan, G., Roy, K.: Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13558–13567 (2020)
14. Hao, Z., Bu, T., Ding, J., Huang, T., Yu, Z.: Reducing ann-snn conversion error through residual membrane potential. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 11–21 (2023)
15. Hao, Z., Ding, J., Bu, T., Huang, T., Yu, Z.: Bridging the gap between anns and snns by calibrating offset spikes. In: International Conference on Learning Representations (2023)

16. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
17. Ikegawa, S.i., Saiin, R., Sawada, Y., Natori, N.: Rethinking the role of normalization and residual blocks for spiking neural networks. Sensors **22**(8), 2876 (2022)
18. Imam, N., Cleland, T.A.: Rapid online learning and robust recall in a neuromorphic olfactory circuit. Nature Machine Intelligence **2**(3), 181–191 (2020)
19. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. pmlr (2015)
20. Jiang, H., Anumasa, S., De Masi, G., Xiong, H., Gu, B.: 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)
21. Jiang, H., Zoonekynd, V., De Masi, G., Gu, B., Xiong, H.: Tab: Temporal accumulated batch normalization in spiking neural networks. In: International Conference on Learning Representations (2024)
22. Kim, Y., Panda, P.: Revisiting batch normalization for training low-latency deep spiking neural networks from scratch. Frontiers in neuroscience **15**, 773954 (2021)
23. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Master's thesis, University of Tront (2009)
24. Li, H., Liu, H., Ji, X., Li, G., Shi, L.: Cifar10-dvs: an event-stream dataset for object classification. Frontiers in neuroscience **11**, 244131 (2017)
25. Li, Y., Guo, Y., Zhang, S., Deng, S., Hai, Y., Gu, S.: Differentiable spike: Rethinking gradient-descent for training spiking neural networks. Advances in Neural Information Processing Systems **34**, 23426–23439 (2021)
26. Ma, R., Meng, L., Qiao, G., Ning, N., Liu, Y., Hu, S.: I2e: Real-time image-to-event conversion for high-performance spiking neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 40, pp. 1982–1990 (2026)
27. Ma, R., Qiao, G., Liu, Y., Meng, L., Ning, N., Liu, Y., Hu, S.: A&b bnn: Add&bit-operation-only hardware-friendly binary neural network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5704–5713 (2024)
28. Merolla, P.A., Arthur, J.V., Alvarez-Icaza, R., Cassidy, A.S., Sawada, J., Akopyan, F., Jackson, B.L., Imam, N., Guo, C., Nakamura, Y., et al.: A million spiking-neuron integrated circuit with a scalable communication network and interface. Science **345**(6197), 668–673 (2014)
29. Neftci, E.O., Mostafa, H., Zenke, F.: 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)
30. Pei, J., Deng, L., Song, S., Zhao, M., Zhang, Y., Wu, S., Wang, G., Zou, Z., Wu, Z., He, W., et al.: Towards artificial general intelligence with hybrid tianjic chip architecture. Nature **572**(7767), 106–111 (2019)
31. Qiao, G., Ning, N., Zuo, Y., Zhou, P., Sun, M., Hu, S., Yu, Q., Liu, Y.: Batch normalization-free weight-binarized snn based on hardware-saving if neuron. Neurocomputing **544**, 126234 (2023)
32. Qiao, N., Mostafa, H., Corradi, F., Osswald, M., Stefanini, F., Sumislawska, D., Indiveri, G.: A reconfigurable on-line learning spiking neuromorphic processor comprising 256 neurons and 128k synapses. Frontiers in neuroscience **9**, 141 (2015)

33. Qiao, S., Wang, H., Liu, C., Shen, W., Yuille, A.: Micro-batch training with batch-channel normalization and weight standardization. arXiv preprint arXiv:1903.10520 (2019)
34. Roy, K., Jaiswal, A., Panda, P.: Towards spike-based machine intelligence with neuromorphic computing. Nature **575**(7784), 607–617 (2019)
35. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International journal of computer vision **115**, 211–252 (2015)
36. Sengupta, A., Ye, Y., Wang, R., Liu, C., Roy, K.: Going deeper in spiking neural networks: Vgg and residual architectures. Frontiers in neuroscience **13**, 95 (2019)
37. Shrestha, S.B., Orchard, G.: Slayer: Spike layer error reassignment in time. Advances in neural information processing systems **31** (2018)
38. Turrigiano, G.G.: Homeostatic plasticity in neuronal networks: the more things change, the more they stay the same. Trends in neurosciences **22**(5), 221–227 (1999)
39. Turrigiano, G.G., Nelson, S.B.: Homeostatic plasticity in the developing nervous system. Nature reviews neuroscience **5**(2), 97–107 (2004)
40. Wang, B., Cao, J., Chen, J., Feng, S., Wang, Y.: A new ann-snn conversion method with high accuracy, low latency and good robustness. In: IJCAI. pp. 3067–3075 (2023)
41. Wu, Y., Deng, L., Li, G., Zhu, J., Shi, L.: Spatio-temporal backpropagation for training high-performance spiking neural networks. Frontiers in neuroscience **12**, 331 (2018)
42. Wu, Y., Deng, L., Li, G., Zhu, J., Xie, Y., Shi, L.: Direct training for spiking neural networks: Faster, larger, better. In: Proceedings of the AAAI conference on artificial intelligence. vol. 33, pp. 1311–1318 (2019)
43. Xiao, M., Meng, Q., Zhang, Z., He, D., Lin, Z.: Online training through time for spiking neural networks. Advances in neural information processing systems **35**, 20717–20730 (2022)
44. Xu, Q., Qi, Y., Yu, H., Shen, J., Tang, H., Pan, G., et al.: Csnn: an augmented spiking based framework with perceptron-inception. In: IJCAI. vol. 1646 (2018)
45. Yun, S., Han, D., Oh, S.J., Chun, S., Choe, J., Yoo, Y.: Cutmix: Regularization strategy to train strong classifiers with localizable features. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6023–6032 (2019)
46. Zenke, F., Bohté, S.M., Clopath, C., Comşa, I.M., Göltz, J., Maass, W., Masquelier, T., Naud, R., Neftci, E.O., Petrovici, M.A., et al.: Visualizing a joint future of neuroscience and neuromorphic engineering. Neuron **109**(4), 571–575 (2021)
47. Zhang, H.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (2018)
48. Zheng, H., Wu, Y., Deng, L., Hu, Y., Li, G.: Going deeper with directly-trained larger spiking neural networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 11062–11070 (2021)
49. Zhou, S., Li, X., Chen, Y., Chandrasekaran, S.T., Sanyal, A.: Temporal-coded deep spiking neural network with easy training and robust performance. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 11143–11151 (2021)
50. Zhou, Z., Zhu, Y., He, C., Wang, Y., YAN, S., Tian, Y., Yuan, L.: Spikformer: When spiking neural network meets transformer. In: The Eleventh International Conference on Learning Representations (2023)