MI-TRQR: Mutual Information-Based Temporal Redundancy Quantification and Reduction for Energy-Efficient Spiking Neural Networks

Dengfeng Xue[1]* **Wenjuan Li**[2,3]† **Yifan Lu**[2,3] **Chunfeng Yuan**[2,3,4] **Yufan Liu**[2,3] **Wei Liu**[2,3]
Man Yao[2] **Li Yang**[2,3] **Guoqi Li**[2] **Bing Li**[2,3,4,5] **Stephen Maybank**[6] **Weiming Hu**[2,3,4,7] **Zhetao Li**[8]
[1]School of Computer Science and Technology, Xidian University
[2]State Key Laboratory of Multimodal Artificial Intelligence Systems, CASIA
[3]Beijing Key Laboratory of Super Intelligent Security of Multi-Modal Information
[4]School of Artificial Intelligence, University of Chinese Academy of Sciences [5]PeopleAI Inc.
[6]Department of Computer Science and Information Systems, Birkbeck College, University of London
[7]School of Information Science and Technology, ShanghaiTech University
[8]College of Information Science and Technology, Jinan University

Abstract

Brain-inspired spiking neural networks (SNNs) provide energy-efficient computation through event-driven processing. However, the shared weights across multiple timesteps lead to serious temporal feature redundancy, limiting both efficiency and performance. This issue is further aggravated when processing static images due to the duplicated input. To mitigate this problem, we propose a parameter-free and plug-and-play module named Mutual Information-based Temporal Redundancy Quantification and Reduction (MI-TRQR), constructing energy-efficient SNNs. Specifically, Mutual Information (MI) is properly introduced to quantify redundancy between discrete spike features at different timesteps on two spatial scales: pixel (local) and the entire spatial features (global). Based on the multi-scale redundancy quantification, we apply a probabilistic masking strategy to remove redundant spikes. The final representation is subsequently recalibrated to account for the spike removal. Extensive experimental results demonstrate that our MI-TRQR achieves sparser spiking firing, higher energy efficiency, and better performance concurrently with different SNN architectures in tasks of neuromorphic data classification, static data classification, and time-series forecasting. Notably, MI-TRQR increases accuracy by **1.7%** on CIFAR10-DVS with 4 timesteps while reducing energy cost by **37.5%**. Our codes are available at `https://github.com/dfxue/MI-TRQR`.

1 Introduction

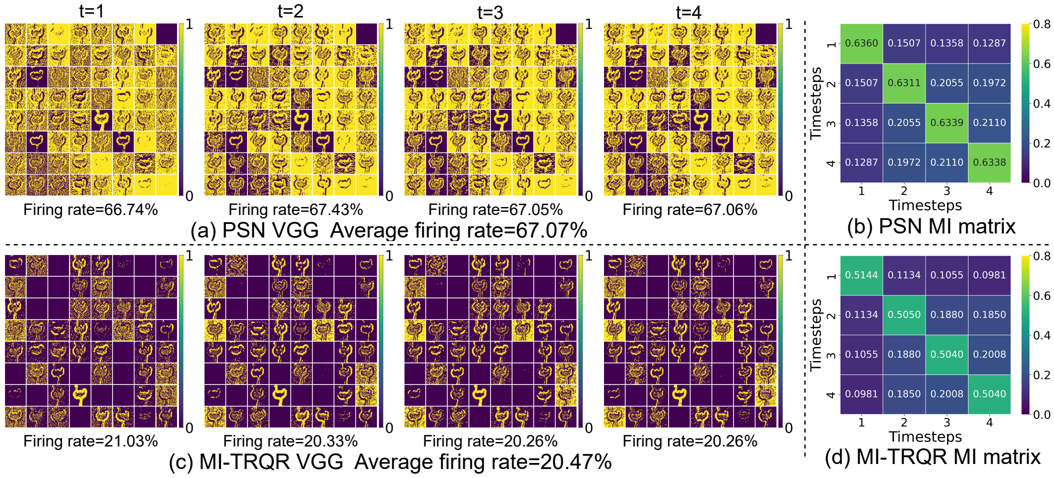
Spiking Neural Networks (SNNs), inspired by the processing mechanisms of biological neural networks [18], offer an energy-efficient approach to neuromorphic hardware by performing spike-based accumulation and avoiding the computation of zero-value inputs (i.e., event-driven) [36, 48, 10]. With the release of neuromorphic chips [5, 37] and the proposal of algorithms for various tasks [56, 11, 87, 79, 50, 63, 73, 64, 55], neuromorphic computing systems are progressing toward practical deployment in real-world applications [30].

SNNs rely on sequential timesteps to transmit temporal information, and their outputs are typically averaged across time to improve accuracy [8, 77]. The temporally shared weights induce spatio-

[*] An intern at CASIA.

[†] Corresponding author: wenjuan.li@ia.ac.cn

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) PSN VGG Average firing rate=67.07%

(b) PSN MI matrix

(c) MI-TRQR VGG Average firing rate=20.47%

(d) MI-TRQR MI matrix

Figure 1: Visualization of spike features on CIFAR10-DVS: (a) Spike features in PSN [14] and (b) their MI matrix; (c) Spike features in MI-TRQR and (d) their MI matrix. MI-TRQR reduces the spiking firing rate by 69.48% (67.07% → 20.47%) and temporal redundancy by about 24% (computed from the redundancy reduction between the first timestep and other timesteps, e.g., 0.1507→0.1134).

temporal invariance [22], introducing serious feature redundancy and resulting in a high spiking firing rate [68, 25], as shown in Fig. 1(a). This phenomenon degrades energy efficiency and compromises the compactness of learned features. When processing static images, existing methods often duplicate the same input across all timesteps [12, 20], further exacerbating temporal redundancy in SNNs [68]. This practice has been shown to impair both the energy efficiency and overall performance of SNNs [48, 38, 69].

Energy efficiency has been a longstanding concern in recent research [54, 49, 46, 4], motivating the research and development of diverse strategies aimed at reducing redundancy. Kim et al. [25] found that features at later timesteps had minimal impact on the final predictions, highlighting considerable temporal redundancy. Yao et al. [68] conducted a systematic yet qualitative analysis on redundancy. Qin et al. [44] provided a quantitative yet indirect strategy by defining similar spikes through the cosine similarity of their membrane potential. We argue that it is insufficient to quantify the redundancy between the spike features due to two intrinsic faults: (1) the inability of a linear tool to capture correlations between floating-point membrane potentials, and (2) the quantization error caused by the discontinuous and nonlinear property of the spiking activation function. The absence of direct redundancy quantification significantly hinders progress toward highly energy-efficient SNNs.

To directly and accurately quantify redundancy between discrete spike features, we propose to use Mutual Information (MI), which is a principled and widely used metric [43, 81, 82, 57]. Existing metrics, such as Pearson correlation and Euclidean distance, measure basic similarity but fall short in capturing the complex and non-linear similarity between high-dimensional spike features [40]. In contrast, Mutual Information (MI) captures complex statistical dependencies through probability distribution analysis, making it well-suited for quantifying redundancy between the high-dimensional discrete spike features. In this work, we calculate MI between spike features at different timesteps, as shown in Fig. 1(b) and (d).

Previous studies analyzing neural recordings from various monkey brain regions have shown that spike features are significantly less redundant with activity-dependent depression [15]. Predictive coding has also demonstrated the ability to learn efficient visual representations by removing pixel-wise redundant spikes [2, 39]. Inspired by these works, we propose a parameter-free, plug-and-play MI-based Temporal Redundancy Quantification and Reduction (MI-TRQR) module, which is seamlessly integrated into the SNNs. Specifically, MI is used to quantify temporal redundancy between high-level spike features on pixel-level (local) and the entire spatial scale (global). These multi-scale redundancy quantifications are aggregated to compute a pixel-wise probability for removing spikes, which is achieved using a binary mask. Existing SNNs typically produce final representations by averaging outputs across timesteps, whether for classification [9, 12, 20] or forecasting [35], implicitly assuming a uniform temporal contribution or information distribution. However, due to the non-uniform information distribution after spike removal, we recalibrate the weights for the final representation. As illustrated in Fig. 1, MI-TRQR reduces spiking firing rate and temporal redundancy. The proposed MI-TRQR is validated on neuromorphic data classification, static image classification, and time-series forecasting tasks. Experimental results demonstrate that our MI-TRQR obtains more compact representations, improving both performance and energy efficiency. Our contributions are as follows:

- We use MI to directly and accurately quantify temporal redundancy between high-dimensional discrete spike features at multiple spatial scales in SNNs.
- We propose a parameter-free, plug-and-play MI-TRQR module, which removes pixel-wise redundant spikes based on the multi-scale redundancy quantification. The weight of the final representation is recalibrated to balance the information distribution.
- We demonstrate the significant advantages of our approach by comparing the energy consumption of MI-TRQR with baseline methods, showing a clear improvement. Extensive experiments across a range of tasks confirm that our method achieves higher accuracy and enhanced energy efficiency.

2 Related Work

Redundancy is a critical factor that must be addressed when designing efficient SNNs. In the following, we briefly review some representative studies that focus on analyzing and/or reducing redundancy in SNNs.

2.1 Redundancy Analysis

Kim et al. [25] investigated the temporal information distribution and identified the Temporal Information Concentration (TIC) phenomenon, wherein information is highly concentrated in the early timesteps after training. This observation highlights the presence of serious temporal redundancy. Yao et al. [68] conducted a systematic analysis, particularly for event-based vision tasks. They discussed three key questions ('which', 'why', and 'how') and concluded that redundancy primarily stems from the spatio-temporal invariance caused by temporally shared weights in SNNs [22]. Qin et al. [44] introduced the concept of a spike cluster, defined based on the cosine similarity of membrane potentials across timesteps. Despite substantial progress in redundancy-related research in recent years, direct quantification of redundancy between spike features across timesteps remains largely unexplored. In this paper, we address this gap by using MI to directly quantify the redundancy between spike features at different timesteps.

2.2 Redundancy Reduction

Many methods have been proposed to mitigate redundancy in different ways. Perez et al. [42] introduced an early sparse backpropagation algorithm tailored for SNNs. Subsequent works further applied sparsity regularization during backpropagation to enhance the energy efficiency of SNNs [66, 75]. A recent series of studies [72, 71, 67, 68, 65] presented a cohesive exploration of attention mechanisms in SNNs. These studies used diverse attention designs to strategically modulate membrane potential distributions and spiking responses across various dimensions, such as spatial, temporal, channel-wise, and/or their combinations. They underscored the transformative impact of attention mechanisms in SNNs, leading to more efficient and accurate models. However, such attention modules inevitably increase network complexity and introduce additional multiplications, which may raise hardware costs and hinder deployment on resource-limited edge devices. The temporal self-erasing method dynamically adjusted the regions of interest for different timesteps [34]. Another widely adopted approach is to apply penalty functions to encourage the deep network to learn sparser spike representations [41, 74, 27, 78]. These methods often require careful tuning of many hyperparameters, which may increase the training complexity. In this paper, we develop a parameter-free and plug-and-play module to enable SNNs to learn compact and powerful representations.

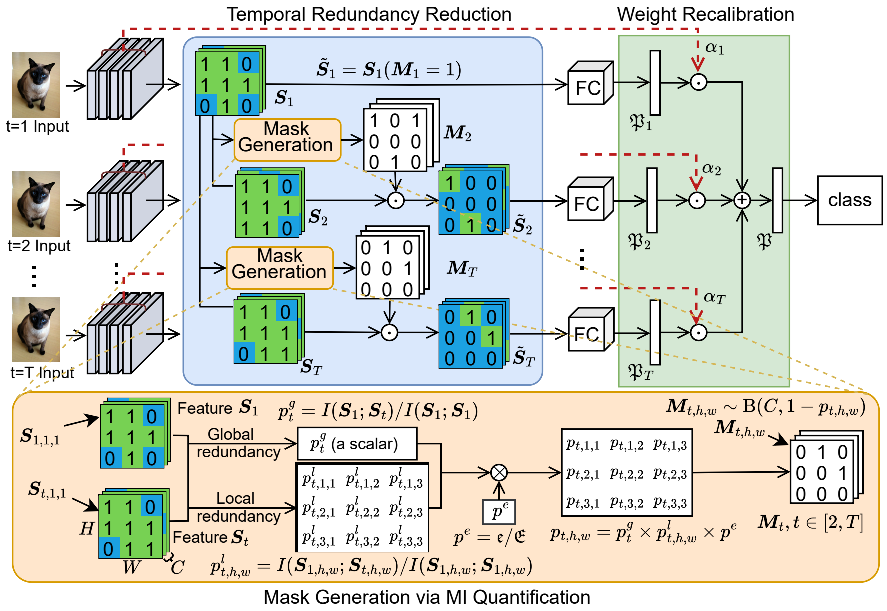

Temporal Redundancy Reduction
Weight Recalibration
t=1 Input
t=2 Input
t=T Input
$\tilde{S}_1 = S_1(M_1 = 1)$
S_1
Mask Generation
M_2
S_2
$\tilde{S}_2$
Mask Generation
M_T
S_T
$\tilde{S}_T$
FC
FC
FC
α_1
α_2
α_T
$\mathfrak{P}_1$
$\mathfrak{P}_2$
$\mathfrak{P}_T$
$\mathfrak{P}$
class
$S_{1,1,1}$
Feature S_1
$p_t^g = I(S_1; S_t)/I(S_1; S_1)$
Global redundancy
p_t^g (a scalar)
$S_{t,1,1}$
Local redundancy
H
W
C
Feature S_t
$p^l_{t,1,1}$ $p^l_{t,1,2}$ $p^l_{t,1,3}$
$p^l_{t,2,1}$ $p^l_{t,2,2}$ $p^l_{t,2,3}$
$p^l_{t,3,1}$ $p^l_{t,3,2}$ $p^l_{t,3,3}$
p^e
$p^e = \mathfrak{e}/\mathfrak{E}$
$p_{t,1,1}$ $p_{t,1,2}$ $p_{t,1,3}$
$p_{t,2,1}$ $p_{t,2,2}$ $p_{t,2,3}$
$p_{t,3,1}$ $p_{t,3,2}$ $p_{t,3,3}$
$p_{t,h,w} = p_t^g \times p^l_{t,h,w} \times p^e$
$M_{t,h,w} \sim \mathrm{B}(C, 1 - p_{t,h,w})$
$M_{t,h,w}$
$M_t, t \in [2, T]$
$p^l_{t,h,w} = I(S_{1,h,w}; S_{t,h,w})/I(S_{1,h,w}; S_{1,h,w})$
Mask Generation via MI Quantification

Figure 2: Our MI-TRQR module preserves the original spike features at the first timestep. At subsequent timesteps, local and global redundancy quantification are first aggregated with a training factor. This aggregation is then used to compute a probability that guides the mask generation. The generated mask is responsible for removing pixel-wise redundant spikes, aiming to achieve temporal redundancy reduction. Weight recalibration balances the non-uniform information distribution.

3 Methodology

In this section, we introduce the proposed MI-TRQR module, which is integrated after the final stage to process the high-level features [83], as shown in Fig. 2. Section 3.1 details how MI is used to quantify temporal redundancy between spike features. Section 3.2 introduces the temporal redundancy reduction strategy, removing redundant spikes using the multi-scale redundancy quantification. Section 3.3 recalibrates the weight of the final representation based on the information density.

3.1 Temporal Redundancy Quantification

MI is particularly effective for evaluating similarity between discrete variables, as elaborated in Appendix A. Therefore, MI is a suitable metric to quantify temporal redundancy between binary spike features. Given the four-dimensional binary spike feature $\boldsymbol{S} \in \{0,1\}^{T \times C \times H \times W}$ where T, C, H, W indicate the timestep, channel, height, and width respectively, we define two terms to quantify temporal redundancy.

Definition 1. *Global redundancy* quantifies the overall temporal redundancy between spike features at two different timesteps and is denoted as R^g. It is computed as follows:

$$R^g(i,j) = I(\boldsymbol{S}_i; \boldsymbol{S}_j) = \sum_{s_i \in \boldsymbol{S}_i} \sum_{s_j \in \boldsymbol{S}_j} p(s_i, s_j) \log \left(\frac{p(s_i, s_j)}{p(s_i)p(s_j)} \right), \tag{1}$$

where $I(\boldsymbol{S}_i; \boldsymbol{S}_j)$ denote MI between spiking features $\boldsymbol{S}_i, \boldsymbol{S}_j \in \{0,1\}^{C \times H \times W}$ at timestep i and j. s_i and s_j represent specific values of $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$. $p(s_i)$ and $p(s_j)$ are the marginal probability distributions of s_i and s_j, respectively. $p(s_i, s_j)$ is their joint probability distribution. $R^g(i,j) \geq 0$ is a float scalar that quantifies the degree of *global redundancy* between spike features $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$.

Definition 2. *Local redundancy* quantifies temporal redundancy at the pixel scale. For each spatial location, we compute the MI between the local spike vectors at two timesteps:

$$R^l_{h,w}(i,j) = I(\boldsymbol{S}_{i,h,w}; \boldsymbol{S}_{j,h,w}) = \sum_{s_i \in \boldsymbol{S}_{i,h,w}} \sum_{s_j \in \boldsymbol{S}_{j,h,w}} p(s_i, s_j) \log \left(\frac{p(s_i, s_j)}{p(s_i)p(s_j)} \right), \tag{2}$$

where $\forall h \in [1, H], \forall w \in [1, W]$. $\boldsymbol{S}_{i,h,w}, \boldsymbol{S}_{j,h,w} \in \{0, 1\}^C$ are the raw pixel (spatial location (h, w)) at timestep i and j, respectively. $R^l_{h,w}(i, j) \geq 0$ is also a float scalar, indicating the pixel-wise redundancy between $\boldsymbol{S}_{i,h,w}$ and $\boldsymbol{S}_{j,h,w}$. It corresponds to the (h, w)-th element of the *local redundancy* matrix $R^l(i, j) \in \mathbb{R}^{H \times W}$.

3.2 Temporal Redundancy Reduction

Based on the redundancy quantification between spike features at multiple spatial scales, we derive a probability that estimates the likelihood of each pixel-wise spike being redundant. This probability is then used to generate a pixel-wise temporal redundancy mask, which selectively removes redundant spikes.

Temporal redundancy-guided probability derivation. The *global redundancy* between spike features $\boldsymbol{S}_1$ and $\boldsymbol{S}_t$ ($t \in [2, T]$) is served as the global probability factor for spikes removal, denoted by $p^g_t = R^g(1, t)$. We observe that the MI of a feature with itself $I(\boldsymbol{S}_1, \boldsymbol{S}_1)$ is not equal to one. This is evident from the varying diagonal elements in the MI matrix shown in Fig. 1. Therefore, we normalize the global probability factor p^g_t as follows:

$$p^g_t = \frac{R^g(1, t)}{R^g(1, 1)} = \frac{I(\boldsymbol{S}_1; \boldsymbol{S}_t)}{I(\boldsymbol{S}_1; \boldsymbol{S}_1)}, \tag{3}$$

where $p^g_t \in [0, 1)$, since $I(\boldsymbol{S}_1; \boldsymbol{S}_t) < I(\boldsymbol{S}_1; \boldsymbol{S}_1)$ when $t \neq 1$, as shown in the Fig 1(b).

Similarly to the computation of p^g_t in Eq. 3, the local probability factor $p^l_{t,h,w}$ is defined as follows:

$$p^l_{t,h,w} = \frac{R^l_{h,w}(1, t)}{R^l_{h,w}(1, 1)} = \frac{I(\boldsymbol{S}_{1,h,w}; \boldsymbol{S}_{t,h,w})}{I(\boldsymbol{S}_{1,h,w}; \boldsymbol{S}_{1,h,w})}, \tag{4}$$

where $p^l_{t,h,w} \in [0, 1)$, similar to the p^g_t.

In addition, the TIC phenomenon demonstrates that information gradually concentrates on the first timestep as training goes on [25]. Accordingly, we retain the spikes at the first timestep, as illustrated in Fig. 2. However, when removing spikes, it is crucial to consider the training epoch, as the informative spikes are still distributed across the later timesteps during the early training stages. Thus, we introduce a training-dependent factor p^e into the pixel-wise probability $p_{t,h,w}$:

$$p_{t,h,w} = p^g_t \times p^l_{t,h,w} \times p^e, \quad p^e = \mathfrak{e}/\mathfrak{E}, \tag{5}$$

where $\mathfrak{e}$ and $\mathfrak{E}$ denote the current training epoch and the total number of epochs, respectively.

Temporal redundancy-based mask. We use the pixel-wise probability $p_{t,h,w}$ to generate a binary mask at timestep t:

$$\boldsymbol{M}_t = \begin{cases} \{1\}^{C \times H \times W} & \text{if } t = 1 \\ [\boldsymbol{M}_{t,h,w}], & \text{otherwise} \end{cases}, \quad \boldsymbol{M}_{t,h,w} \sim \mathrm{B}(C, 1 - p_{t,h,w}), \tag{6}$$

where $\boldsymbol{M}_t \in \{0, 1\}^{C \times H \times W}$ denotes the mask at timestep t. $\boldsymbol{M}_{t,h,w} \in \{0, 1\}^C$ represents the pixel at spatial location (h, w). $\mathrm{B}(\cdot, \cdot)$ denotes the Binomial distribution. The spike removal operation using the mask $\boldsymbol{M}_t$ is expressed as:

$$\tilde{\boldsymbol{S}}_t = \boldsymbol{S}_t \odot \boldsymbol{M}_t, \tag{7}$$

where $\tilde{\boldsymbol{S}}_t$ denotes the output spike features at timestep t, and $\odot$ is the element-wise multiplication. The mask at the first timestep, $\boldsymbol{M}_1$, consists entirely of one, thereby preserving the original spikes.

Algorithm 1 The implementation of removing pixel-wise redundant spikes

```
Input: Spike feature $\boldsymbol{S} \in \{0,1\}^{T \times C \times H \times W}$ , the training factor $p^e$
Output: Spike feature $\tilde{\boldsymbol{S}} \in \{0,1\}^{T \times C \times H \times W}$
Obtain the number of spatial pixels: $N = H \times W$
Flatten spatial dimension: $\boldsymbol{S'}$ = reshape $(\boldsymbol{S}, (T, C, N))$
Define a matrix filled with 1 for redundancy quantification: $\mathfrak{R}' \leftarrow$ ones $(T, N)$
for $t = 1 \to T$ do
    Compute *global redundancy*: $R^g(1,t) = I(\boldsymbol{S'_1}; \boldsymbol{S'_t})$
    for $n = 1 \to N$ do
        Compute *local redundancy*: $R^l_n(1,t) = I(\boldsymbol{S'_{1,n}}; \boldsymbol{S'_{t,n}})$,
        Get the pixel-wise combined redundancy metric: $\mathfrak{R}'[t,n] = R^g(1,t) \times R^l_n(1,t)$
    end for
end for
Unfold spatial dimension: $\mathfrak{R}$ = reshape $(\mathfrak{R}', (T, H, W))$
Obtain a $T \times N$ probability matrix: $\boldsymbol{P} = \mathfrak{R}[:,:,:]/\mathfrak{R}[0, ; , :] \times p^e$
Set the probability at the first timestep into zeros: $\boldsymbol{P}[0]$=0
Generate mask: $\boldsymbol{M} = [\boldsymbol{M}_{t,h,w}], \quad \boldsymbol{M}_{t,h,w} \sim \mathrm{B}(C, 1 - \boldsymbol{P}[t,h,w])$,
Get the output spike feature with element-wise multiplication: $\tilde{\boldsymbol{S}} = \boldsymbol{S} \odot \boldsymbol{M}$
```

For $t \in [2, T]$, zero values in $\boldsymbol{M}_t$ convert active spikes (1) into inactive states (0). Consequently, a proportion ($p_{t,h,w}$) of spikes at the pixel $\boldsymbol{S}_{t,h,w}$ is removed. The pseudo-code for spike removal is given in Alg. 1. Spike removal also reduces the spiking firing rate in earlier layers, which is analyzed in Appendix B.

3.3 Weight Recalibration

The spike removal at later timesteps introduces a significant divergence in information distribution across timesteps. Consequently, it is unsuitable to simply average temporal outputs for the final representation, as done in existing works [12, 20, 35]. To address this issue, we adaptively recalibrate the weight of the final representation using the normalized network spiking firing rate:

$$\alpha_t = \frac{fr^{net}_t}{\sum_{\tau=1}^{T} fr^{net}_\tau}, \quad fr^{net}_t = \frac{\sum_{l=1}^{L} N^s_{l,t}}{\sum_{l=1}^{L} N^e_{l,t}}, \tag{8}$$

where a_t and fr^{net}_t denote the weight and network spiking firing rate at timestep t, respectively. $N^s_{l,t}$ indicates the number of spikes, and $N^e_{l,t}$ indicates the total number of elements, both at timestep t and layer l. L is the total number of layers. In this way, the final representation $\mathfrak{P}$ is obtained as follows:

$$\mathfrak{P} = \sum_{t=1}^{T} (\alpha_t \times \mathfrak{P}_t), \tag{9}$$

where $\mathfrak{P}_t$ denotes the representation at timestep t.

4 Experiments

In this section, we report the experimental results for MI-TRQR on both classification and time-series forecasting tasks. The operation of removing spikes is only used during training. The experimental setup is described in Appendix C.1.

4.1 Comparison with Other Methods

Neuromorphic data classification. On CIFAR10-DVS, MI-TRQR consistently surpasses PSN [14] in both accuracy and energy efficiency across different timesteps, as shown in Tab. 1. When T=4, PSN vs. **MI-TRQR**: Accuracy, 82.3% vs. **83.9%(+1.6%)**; Power, 0.72mJ vs. **0.45mJ (-0.27mJ,**

Table 1: Comparison with other methods on neuromorphic datasets. Power denotes the energy cost of inference one test sample. ↑ and ↓ indicate desired metric direction. Results denoted by ♠ were obtained through our reimplementation.

Dataset	Model	Network	T	Accuracy (↑%)	Power (↓mJ)
CIFAR10-DVS	DeepTAGE [33]	VGG-11	10	81.2	-
	TIM [53]	Spikformer	10	81.6	-
	SEMM [85]	Spikformer-2-256	16	82.9	-
	RM-SNN [71]	VGG	10	82.9	-
	STAtten [29]	SpikingReformer-4-384	16	83.9	-
	QKFormer [84]	HST-2-256	16	84.0	-
	CLIF [21]	VGG	10	86.1	-
	PSN [14]	VGG	4	82.3	0.72
			8	85.3	1.28
			10	85.9	1.20
	MI-TRQR (ours)	VGG	4	**83.9**±0.05	**0.45**±0.06
			8	**86.2**±0.12	**0.74**±0.07
			10	**86.5**±0.13	**0.84**±0.07
Gait	ASA [68]	3-Layer SNN	10	83.2	-
	3D GCN [60]	-	1	86.0	-
	DSNN [70]	4-Layer SNN	-	90.2	-
	PSN [14]	3B-Net	10	88.8♠	0.19
	MI-TRQR (ours)	3B-Net	10	**90.6**±0.13	**0.13**±0.08

Table 2: Comparison with other methods on ImageNet.

Method	Model	Network	T	Accuracy (↑%)	Power (↓mJ)
ANN2SNN	Hybird [47]	ResNet34	250	61.48	-
	Tandem [62]	VGG-16	16	65.08	-
	Two-stage [61]	ResNet34	16	67.77	-
	Optimal [3]	ResNet34	32	69.37	-
Direct Training	OSR+OTS [89]	ResNet34	4	67.54	-
	EnOF-SNN [17]	ResNet34	4	67.40	-
	rate$_M$ [76]	MS-ResNet34	4	70.01	-
	GAC-SNN [45]	MS-ResNet34	6	70.42	3.38
	IMP+LTS [52]	SEW-ResNet50	4	71.83	3.11
	PSN [14]	SEW-ResNet18	4	67.63	2.42
		SEW-ResNet34	4	70.54	3.70
		SEW-ResNet50	4	72.01♠	4.11
Direct Training	MI-TRQR (ours)	SEW-ResNet18	4	**68.28**±0.12	**1.92**±0.07
		SEW-ResNet34	4	**71.06**±0.11	**3.06**±0.09
		SEW-ResNet50	4	**73.23**±013	**3.09**±0.08

↓37.50%). Similar advantages of our MI-TRQR are observed when T=8,10. MI-TRQR shows 1.8% accuracy improvement and 31.6% energy reduction over PSN on Gait (T=10).

Static data classification. On ImageNet, MI-TRQR consistently outperforms PSN across different backbones, as shown in Tab. 2. For example, with ResNet50, PSN vs. **MI-TRQR**: Accuracy, 72.01% vs. **73.23% (+1.22%)**; Power, 4.11mJ vs. **3.09mJ (-1.02mJ, ↓ 24.82%)**. More experiments on CIFAR10/100 are provided in Appendix C.2.1. On CIFAR10, PSN vs. **MI-TRQR**: Accuracy, 95.32% vs. **95.83% (+0.51%)**; Power, 1.32mJ vs. **0.35mJ (-0.97mJ, ↓ 73.48%)**.

Time-series forecasting. In Tab. 3, we show the superiority of our MI-TRQR over CPG-PE [35] in two metrics (R^2 and RSE) on the Electricity dataset with various prediction lengths 6,24,48,96.

Table 3: Results on Electricity. Our results are averaged across 3 random seeds.

Method	Metric	Eletricity				Avg.
		6	24	48	96	
Spikformer [86]	R^2↑	.956	.955	.953	.943	.952
	RSE ↓	.371	.375	.386	.450	.396
CPG-PE [35]	R^2↑	.971	.971	.968	.962	.968
	RSE ↓	.304	.308	.311	.439	.341
MI-TRQR (ours)	R^2↑	**.973**	**.972**	**.972**	**.967**	**.971**
	RSE ↓	**.292**	**.307**	**.297**	**.368**	**.316**

Table 4: Impact of *local redundancy* (LR) and *global redundancy* (GR) on CIFAR10-DVS.

Method	LR	GR	T	Accuracy (↑%)	Firing rate (↓%)	Training time
PSN [14]	-	-	4	82.3	14.47	**24.1**s
MI-TRQR (ours)	✓	-	4	83.2 (↑0.9)	9.73 (↓32.76)	51.1s
	-	✓	4	83.4 (↑1.1)	10.91 (↓24.60)	24.8s
	✓	✓	4	**84.0** (↑1.7)	**8.35** (↓42.29)	51.9s

Table 5: Impact of MI-TRQR placement on CIFAR10-DVS.

Method	Placement	T	Accuracy(↑%)	Firing rate(↓%)	Training time
PSN [14]	-	4	82.3	14.47	**24.1**s
MI-TRQR(ours)	After layer 4	4	82.8(↑0.5)	11.52(↓20.39%)	597.3s
	After layer 6	4	83.4(↑1.1)	10.18(↓29.65%)	171.6s
	After layer 6+8	4	83.5(↑1.2)	9.74(↓32.69%)	222.6s
	After layer 8(last)	4	**84.0**(↑1.7)	**8.35**(↓42.29%)	**51.9**s

4.2 Ablation Study

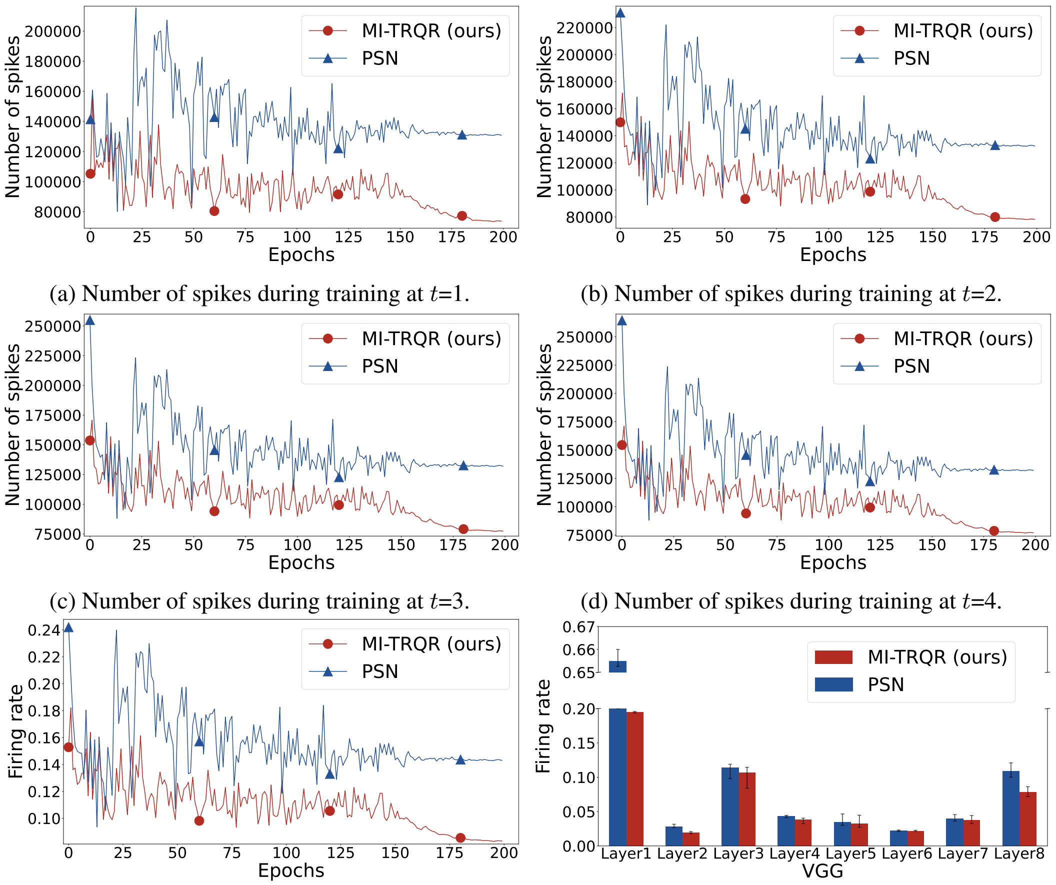
We conducted ablation studies on CIFAR10-DVS to verify the impact of weight recalibration and the effectiveness, efficiency, and optimal placement of the MI-TRQR module. Additional ablation results on CIFAR10 are provided in Appendix C.2.2. More ablation studies are provided in Appendix C.3.

Effectiveness. In Tab. 4, we quantitatively show the gains of incorporating local and/or global redundancy for learning more compact representations. By leveraging multi-scale redundancy, MI-TRQR obtains the best accuracy of 84.0%(+1.7%). We also report the training time per epoch to provide a comprehensive comparison.

Efficiency. In Tab. 4, we can see that using multi-scale redundancy reduces the spiking firing rate by 42.29% (from 14.47% to 8.35%). The detailed spike counts and firing rates are presented in Fig. 3. Key observations are: (1) MI-TRQR consistently fires fewer spikes than PSN at each timestep; (2) its spike removal propagates from deeper layers to shallower layers; (3) it consistently reduces the spiking firing rate across various datasets (CIFAR10 in Appendix C.2.2).

MI-TRQR placement. In Tab. 5, we report the results when MI-TRQR is integrated after different convolutional layers. Results show that positioning MI-TRQR after the last stage (layer 8) yields the biggest advantages in both accuracy and firing rate compared to PSN, confirming the benefit of targeting MI-TRQR to high-level features. In addition, the computation of probability $p_{t,h,w}$ in Eq. 5 is spatiotemporally dependent, with a computational complexity of $O(T)$ for global redundancy and $O(T \times H \times W)$ for local redundancy. Inserting the module into earlier layers leads to a higher computational cost. We provide the corresponding time of training an epoch.

Weight recalibration. Tab. 6 presents the impact of weight recalibration on CIFAR10-DVS. MI-TRQR with reweighting yields the highest accuracy while consuming the least energy.



(a) Number of spikes during training at t=1.

(b) Number of spikes during training at t=2.

(c) Number of spikes during training at t=3.

(d) Number of spikes during training at t=4.

(e) Network spiking firing rate during training (T=4).

(f) Layer-wise spiking firing rate of VGG.

Figure 3: Comparison of the number of spikes and firing rate in different methods on CIFAR10-DVS.

Table 6: Impact of weight recalibration on CIFAR10-DVS.

Method	Reweighting	T	Accuracy (↑%)	Firing rate (↓%)
PSN [14]	-	4	82.3	14.47
MI-TRQR (ours)	-	4	83.4(↑1.1)	9.22(↓36.28)
	✓	4	**84.0**(↑1.7)	**8.35**(↓42.29)

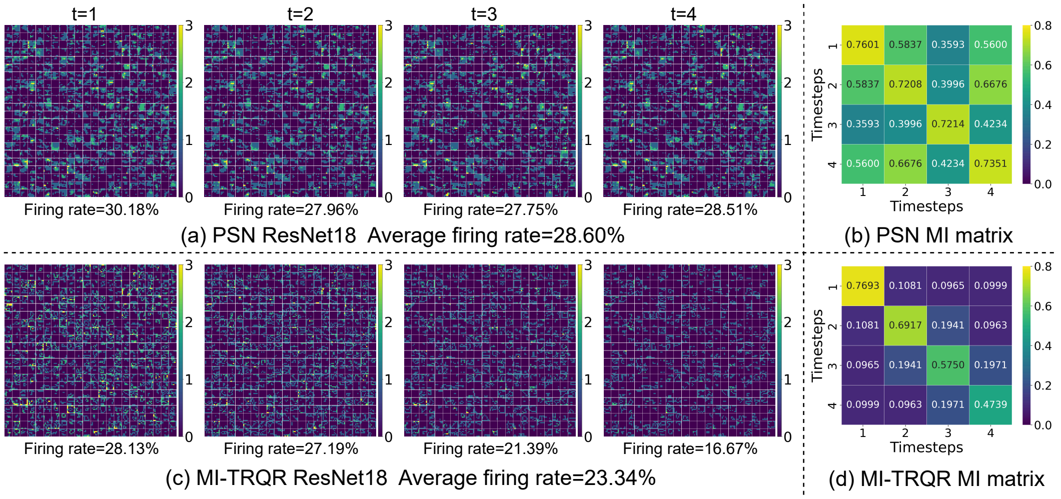
4.3 Visualization

CIFAR10-DVS. In Fig. 1, we can observe that MI-TRQR reduces the spiking firing rate by 69.48% (67.07% → 20.47%) and temporal redundancy by approximately 24% (e.g., 0.1507 →0.1134) even at a shallow layer.

ImageNet. In Fig. 4, we visualize spike features and their MI matrices. We can see that MI-TRQR significantly decreases the temporal redundancy between spike features by 79% (e.g.,) and reduces the average spiking firing rate by 18.39% (23.34% vs. 28.60%). These results demonstrate that MI-TRQR can effectively remove redundant spikes and enhance representation compactness.

5 Conclusion and Limitations

In this paper, we propose an effective and efficient module, named MI-TRQR. Based on the temporal redundancy quantification using MI, MI-TRQR identifies and removes pixel-wise redundant spikes, enabling SNNs to learn more compact and powerful representations. Experimental results on various



(a) PSN ResNet18 Average firing rate=28.60%

(b) PSN MI matrix

(c) MI-TRQR ResNet18 Average firing rate=23.34%

(d) MI-TRQR MI matrix

Figure 4: Visualization of spike features after stage 4 of ResNet18 on ImageNet: (a) Spike features in PSN and (b) their MI matrix; (c) Spike features in MI-TRQR and (d) their MI matrix. For a single sample, the spike features has shape $[T, C, H, W]$ (timestep, channel, height, width). Since $T = 4$, we first split it into four $[C, H, W]$ features (shown on the left). We plot each channel in a small grid. We report the average spiking firing rates under the spike features. MI-TRQR reduces the spiking firing rate by 18.39% (28.60% $\rightarrow$ 23.34%) and temporal redundancy by about 79% (computed from the redundancy reduction between the first timestep and other timesteps, e.g., 0.5837→0.1081)

tasks demonstrate that MI-TRQR consistently outperforms baseline methods in both accuracy and energy efficiency. To the best of our knowledge, this is the first study to use MI to quantify redundancy between spike features in SNNs, providing a foundation for subsequent related research.

Limitations Although our method does not introduce extra cost during inference, it requires computing mutual information between spike features during training, resulting in increased computational cost and longer training time.

Acknowledgements

This work is supported by the Natural Science Foundation of China (62036011, 62192782, W2411053, 62032020, U2441241, U24A20331, 62372451, 62192785, 62372082, 62202469, 62403462, 62202470, 62473363), Beijing Natural Science Foundation (L223003, L243015, L251005, JQ24022), the Key R&D Program of Xinjiang Uyghur Autonomous Region (2023B01005), CAAI-Ant Group Research Fund (CAAI-MYJJ 2024-02), and Young Elite Scientists Sponsorship Program by CAST (2024QNRC001).

References

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

[2] Rafal Bogacz. A tutorial on the free-energy framework for modelling perception and learning. *Journal of Mathematical Psychology*, 76:198–211, 2017.

[3] Tong Bu, Wei Fang, Jianhao Ding, PengLin Dai, Zhaofei Yu, and Tiejun Huang. Optimal ann-snn conversion for high-accuracy and ultra-low-latency spiking neural networks. In *The Tenth International Conference on Learning Representations*, 2022.

[4] Simon Davidson and Steve B Furber. Comparison of artificial and spiking neural networks on digital hardware. *Frontiers in Neuroscience*, 15:651141, 2021.

[5] 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.

[6] 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*, pages 248–255. IEEE, 2009.

[7] Shikuang Deng, Yuhang Wu, Kangrui Du, and Shi Gu. Spiking token mixer: An event-driven friendly former structure for spiking neural networks. In *Advances in Neural Information Processing Systems*, volume 37, pages 128825–128846, 2024.

[8] Yongqi Ding, Lin Zuo, Mengmeng Jing, Pei He, and Hanpu Deng. Rethinking spiking neural networks from an ensemble learning perspective. In *The Thirteenth International Conference on Learning Representations*, 2025.

[9] 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.

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

[11] Bin Fan, Jiaoyang Yin, Yuchao Dai, Chao Xu, Tiejun Huang, and Boxin Shi. Spatio-temporal interactive learning for efficient image reconstruction of spiking cameras. *Advances in Neural Information Processing Systems*, 37:21401–21427, 2024.

[12] 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, 2021.

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

[14] Wei Fang, Zhaofei Yu, Zhaokun Zhou, Ding Chen, Yanqi Chen, Zhengyu Ma, Timothée Masquelier, and Yonghong Tian. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[15] Mark S. Goldman, Pedro Maldonado, and L. F. Abbott. Redundancy reduction and sustained firing with stochastic depressing synapses. *Journal of Neuroscience*, 22(2):584–591, 2002.

[16] Yufei Guo, Xiaode Liu, Yuanpei Chen, Weihang Peng, Yuhan Zhang, and Zhe Ma. Spiking transformer: Introducing accurate addition-only spiking self-attention for transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24398–24408, 2025.

[17] Yufei Guo, Weihang Peng, Xiaode Liu, Yuanpei Chen, Yuhan Zhang, Xin Tong, Zhou Jie, and Zhe Ma. Enof-snn: Training accurate spiking neural networks via enhancing the output feature. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[18] Andreas VM Herz, Tim Gollisch, Christian K Machens, and Dieter Jaeger. Modeling single-neuron dynamics and computations: a balance of detail and abstraction. *Science*, 314(5796):80–85, 2006.

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

[20] Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks toward deep residual learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[21] Yulong Huang, Xiaopeng Lin, Hongwei Ren, Haotian Fu, Yue Zhou, Zunchang Liu, Biao Pan, and Bojun Cheng. Clif: Complementary leaky integrate-and-fire neuron for spiking neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

[22] Ziyuan Huang, Shiwei Zhang, Liang Pan, Zhiwu Qing, Mingqian Tang, Ziwei Liu, and Marcelo H Ang Jr. Tada! temporally-adaptive convolutions for video understanding. In *The Tenth International Conference on Learning Representations, ICLR*, 2022.

[23] Haiyan Jiang, Giulia De Masi, Huan Xiong, and Bin Gu. Ndot: neuronal dynamics-based online training for spiking neural networks. In *Forty-first International Conference on Machine Learning*, 2024.

[24] Haiyan Jiang, Vincent Zoonekynd, Giulia De Masi, Bin Gu, and Huan Xiong. TAB: Temporal accumulated batch normalization in spiking neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

[25] Youngeun Kim, Yuhang Li, Hyoungseob Park, Yeshwanth Venkatesha, Anna Hambitzer, and Priyadarshini Panda. Exploring temporal information dynamics in spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8308–8316, 2023.

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

[27] Souvik Kundu, Gourav Datta, Massoud Pedram, and Peter A Beerel. Spike-thrift: Towards energy-efficient deep spiking neural networks by limiting spiking activity via attention-guided compression. In *Proceedings of the IEEE/CVF Winter Conference on Aplications of Computer Vision*, pages 3953–3962, 2021.

[28] Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 95–104, 2018.

[29] Donghyun Lee, Yuhang Li, Youngeun Kim, Shiting Xiao, and Priyadarshini Panda. Spiking transformer with spatial-temporal attention. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 13948–13958, 2025.

[30] Guoqi Li, Lei Deng, Huajin Tang, Gang Pan, Yonghong Tian, Kaushik Roy, and Wolfgang Maass. Brain-inspired computing: A systematic survey and future trends. *Proceedings of the IEEE*, 2024.

[31] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:244131, 2017.

[32] Qianhui Liu, Jiaqi Yan, Malu Zhang, Gang Pan, and Haizhou Li. Lite-snn: designing lightweight and efficient spiking neural network through spatial-temporal compressive network search and joint optimization. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 3097–3105, 2024.

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

[34] Wei Liu, Li Yang, Mingxuan Zhao, Dengfeng Xue, Shuxun Wang, Boyu Cai, Jin Gao, Wenjuan Li, Bing Li, and Weiming Hu. Towards more discriminative feature learning in snns with temporal-self-erasing supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 1420–1428, 2025.

[35] Changze Lv, Dongqi Han, Yansen Wang, Xiaoqing Zheng, Xuanjing Huang, and Dongsheng Li. Advancing spiking neural networks for sequential modeling with central pattern generators. *Advances in Neural Information Processing Systems*, 37:26915–26940, 2024.

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

[37] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[38] Surya Narayanan, Karl Taht, Rajeev Balasubramonian, Edouard Giacomin, and Pierre-Emmanuel Gaillardon. Spinalflow: An architecture and dataflow tailored for spiking neural networks. In *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pages 349–362. IEEE, 2020.

[39] Antony W N'dri, William Gebhardt, Céline Teulière, Fleur Zeldenrust, Rajesh PN Rao, Jochen Triesch, and Alexander Ororbia. Predictive coding with spiking neural networks: a survey. *arXiv preprint arXiv:2409.05386*, 2024.

[40] 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.

[41] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. Learning to be efficient: Algorithms for training low-latency, low-compute deep spiking neural networks. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 293–298, 2016.

[42] N. Perez-Nieves and D. Goodman. Sparse spiking gradient descent. *Advances in Neural Information Processing Systems*, 34:11975–11808, 2021.

[43] Haina Qin, Wenyang Luo, Libin Wang, Dandan Zheng, Jingdong Chen, Ming Yang, Bing Li, and Weiming Hu. Reversing flow for image restoration. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 7545–7558, 2025.

[44] Lang Qin, Ziming Wang, Rui Yan, and Huajin Tang. Attention-based deep spiking neural networks for temporal credit assignment problems. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):10301–10311, 2023.

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

[46] Arjun Rao, Philipp Plank, Andreas Wild, and Wolfgang Maass. A long short-term memory for ai applications in spike-based neuromorphic hardware. *Nature Machine Intelligence*, 4(5):467–479, 2022.

[47] Nitin Rathi, Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. In *8th International Conference on Learning Representations*, 2020.

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

[49] Catherine D Schuman, Shruti R Kulkarni, Maryam Parsa, J Parker Mitchell, Prasanna Date, and Bill Kay. Opportunities for neuromorphic computing algorithms and applications. *Nature Computational Science*, 2(1):10–19, 2022.

[50] 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.

[51] Claude Elwood Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

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

[53] Sicheng Shen, Dongcheng Zhao, Guobin Shen, and Yi Zeng. Tim: an efficient temporal interaction module for spiking transformer. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 3133–3141, 2024.

[54] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13693–13696, 2020.

[55] Qiaoyi Su, Yuhong Chou, Yifan Hu, Jianing Li, Shijie Mei, Ziyang Zhang, and Guoqi Li. Deep directly-trained spiking neural networks for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6555–6565, 2023.

[56] Qiaoyi Su, Weihua He, Xiaobao Wei, Bo Xu, and Guoqi Li. Multi-scale full spike pattern for semantic segmentation. *Neural Networks*, 176:106330, 2024.

[57] Hanshi Wang, Jin Gao, Weiming Hu, and Zhipeng Zhang. Mambafusion: Height-fidelity dense global fusion for multi-modal 3d object detection. *arXiv preprint arXiv:2507.04369*, 2025.

[58] Shuai Wang, Malu Zhang, Dehao Zhang, Ammar Belatreche, Yichen Xiao, Yu Liang, Yimeng Shan, Qian Sun, Enqi Zhang, and Yang Yang. Spiking vision transformer with saccadic attention. In *The Thirteenth International Conference on Learning Representations*, 2025.

[59] Yanxiang Wang, Bowen Du, Yiran Shen, Kai Wu, Guangrong Zhao, Jianguo Sun, and Hongkai Wen. Ev-gait: Event-based robust gait recognition using dynamic vision sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6358–6367, 2019.

[60] Yanxiang Wang, Xian Zhang, Yiran Shen, Bowen Du, Guangrong Zhao, Lizhen Cui, and Hongkai Wen. Event-stream representation for human gaits identification using deep neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3436–3449, 2021.

[61] Ziming Wang, Yuhao Zhang, Shuang Lian, Xiaoxin Cui, Rui Yan, and Huajin Tang. Toward high-accuracy and low-latency spiking neural networks with two-stage optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[62] Jibin Wu, Chenglin Xu, Xiao Han, Daquan Zhou, Malu Zhang, Haizhou Li, and Kay Chen Tan. Progressive tandem learning for pattern recognition with deep spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):7824–7840, 2021.

[63] Keming Wu, Man Yao, Yuhong Chou, Xuerui Qiu, Rui Yang, Bo XU, and Guoqi Li. RSC-SNN: Exploring the trade-off between adversarial robustness and accuracy in spiking neural networks via randomized smoothing coding. In *ACM Multimedia 2024*, 2024.

[64] Xiaoshan Wu, Weihua He, Man Yao, Ziyang Zhang, Yaoyuan Wang, Bo Xu, and Guoqi Li. Event-based depth prediction with deep spiking neural network. *IEEE Transactions on Cognitive and Developmental Systems*, 16(6):2008–2018, 2024.

[65] Dengfeng Xue, Wenjuan Li, Chunfeng Yuan, Haowei Liu, Man Yao, Wei Liu, Li Yang, Bing Li, Weiming Hu, Haoliang Sun, et al. Msfi: Multi-timescale spatio-temporal features integration in spiking neural networks. *Neural Networks*, page 108024, 2025.

[66] Yulong Yan, Haoming Chu, Yi Jin, Yuxiang Huan, Zhuo Zou, and Lirong Zheng. Backpropagation with sparsity regularization for spiking neural network learning. *Frontiers in Neuroscience*, 16:760298, 2022.

[67] Man Yao, Huanhuan Gao, Guangshe Zhao, Dingheng Wang, Yihan Lin, Zhaoxu Yang, and Guoqi Li. Temporal-wise attention spiking neural networks for event streams classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10221–10230, 2021.

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

[69] Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):2973–2990, 2025.

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

[71] Man Yao, Hengyu Zhang, Guangshe Zhao, Xiyu Zhang, Dingheng Wang, Gang Cao, and Guoqi Li. Sparser spiking activity can be better: Feature refine-and-mask spiking neural network for event-based visual recognition. *Neural Networks*, 166:410–423, 2023.

[72] Man Yao, Guangshe Zhao, Hengyu Zhang, Yifan Hu, Lei Deng, Yonghong Tian, Bo Xu, and Guoqi Li. Attention spiking neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[73] Zhiwei Yao, Shaobing Gao, and Wenjuan Li. Snn using color-opponent and attention mechanisms for object recognition. *Pattern Recognition*, 158:111070, 2025.

[74] Bojian Yin, Federico Corradi, and Sander M Bohté. Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks. *Nature Machine Intelligence*, 3(10):905–913, 2021.

[75] Hang Yin, John Boaz Lee, Xiangnan Kong, Thomas Hartvigsen, and Sihong Xie. Energy-efficient models for high-dimensional spike train classification using sparse spiking neural networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2017–2025, 2021.

[76] Chengting Yu, Lei Liu, Gaoang Wang, Erping Li, and Aili Wang. Advancing training efficiency of deep spiking neural networks through rate-based backpropagation. *Advances in Neural Information Processing Systems*, 2024.

[77] Chengting Yu, Xiaochen Zhao, Lei Liu, Shu Yang, Gaoang Wang, Erping Li, and Aili Wang. Efficient logit-based knowledge distillation of deep spiking neural networks for full-range timestep deployment. In *Forty-second International Conference on Machine Learning*, 2025.

[78] Friedemann Zenke and Tim P Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *Neural Computation*, 33(4):899–925, 2021.

[79] Dehao Zhang, Shuai Wang, Ammar Belatreche, Wenjie Wei, Yichen Xiao, Haorui Zheng, Zijian Zhou, Malu Zhang, and Yang Yang. Spike-based neuromorphic model for sound source localization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[80] Hong Zhang and Yu Zhang. Memory-efficient reversible spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16759–16767, 2024.

[81] Yuhan Zhang, Xiaode Liu, Yuanpei Chen, Weihang Peng, Yufei Guo, Xuhui Huang, and Zhe Ma. Enhancing representation of spiking neural networks via similarity-sensitive contrastive learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16926–16934, 2024.

[82] Zhirun Zheng, Zhetao Li, Hongbo Jiang, Leo Yu Zhang, and Dengbiao Tu. Semantic-aware privacy-preserving online location trajectory data sharing. *IEEE Transactions on Information Forensics and Security*, 17:2256–2271, 2022.

[83] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2921–2929, 2016.

[84] Chenlin Zhou, Han Zhang, Zhaokun Zhou, Liutao Yu, Liwei Huang, Xiaopeng Fan, Li Yuan, Zhengyu Ma, Huihui Zhou, and Yonghong Tian. Qkformer: Hierarchical spiking transformer using qk attention. *Advances in Neural Information Processing Systems*, 37:13074–13098, 2024.

[85] Zhaokun Zhou, Yijie Lu, Yanhao Jia, Kaiwei Che, Jun Niu, Liwei Huang, Xinyu Shi, Yuesheng Zhu, Guoqi Li, Zhaofei Yu, and Li Yuan. Spiking transformer with experts mixture. In *Advances in Neural Information Processing Systems*, volume 37, pages 10036–10059, 2024.

[86] Zhaokun Zhou, Yuesheng Zhu, He Chao, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. In *The Eleventh International Conference on Learning Representations*, 2023.

[87] Rui-Jie Zhu, Ziqing Wang, Leilani Gilpin, and Jason Eshraghian. Autonomous driving with spiking neural networks. *Advances in Neural Information Processing Systems*, 37:136782–136804, 2024.

[88] Rui-Jie Zhu, Malu Zhang, Qihang Zhao, Haoyu Deng, Yule Duan, and Liang-Jian Deng. Tcja-snn: Temporal-channel joint attention for spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2024.

[89] Yaoyu Zhu, Jianhao Ding, Tiejun Huang, Xiaodong Xie, and Zhaofei Yu. Online stabilization of spiking neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

[90] Zhengyang Zhuge, Peisong Wang, Xingting Yao, and Jian Cheng. Towards efficient spiking transformer: a token sparsification framework for training and inference acceleration. In *Forty-first International Conference on Machine Learning*, 2024.

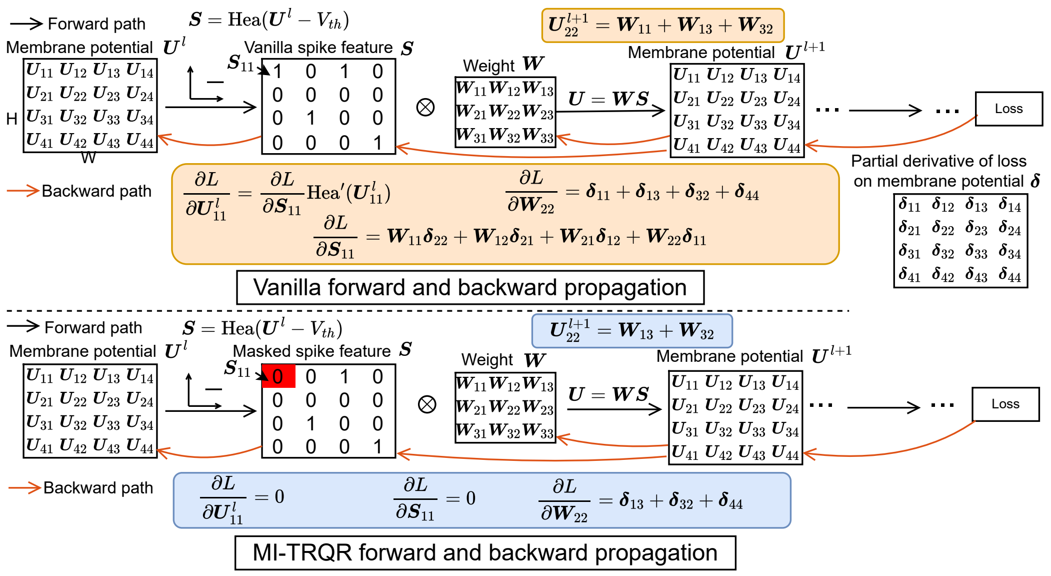

Forward path
$S = \mathrm{Hea}(U^l - V_{th})$
$U_{22}^{l+1} = W_{11} + W_{13} + W_{32}$
Membrane potential U^l
Vanilla spike feature S
Weight W
Membrane potential U^{l+1}
$U = WS$
Loss
H
W
Backward path
$\frac{\partial L}{\partial U_{11}^l} = \frac{\partial L}{\partial S_{11}} \mathrm{Hea}'(U_{11}^l)$
$\frac{\partial L}{\partial W_{22}} = \delta_{11} + \delta_{13} + \delta_{32} + \delta_{44}$
$\frac{\partial L}{\partial S_{11}} = W_{11}\delta_{22} + W_{12}\delta_{21} + W_{21}\delta_{12} + W_{22}\delta_{11}$
Partial derivative of loss on membrane potential δ
Vanilla forward and backward propagation
Forward path
$S = \mathrm{Hea}(U^l - V_{th})$
$U_{22}^{l+1} = W_{13} + W_{32}$
Membrane potential U^l
Masked spike feature S
Weight W
Membrane potential U^{l+1}
$U = WS$
Loss
Backward path
$\frac{\partial L}{\partial U_{11}^l} = 0$
$\frac{\partial L}{\partial S_{11}} = 0$
$\frac{\partial L}{\partial W_{22}} = \delta_{13} + \delta_{32} + \delta_{44}$
MI-TRQR forward and backward propagation

Figure 5: Forward and backward process in vanilla and MI-TRQR method. We detail the forward and backward propagation processes of a layer of the SNN network, only considering the calculation in the spatial domain. When performing convolution calculations, the padding (without drawing) and stride are set to 1. A masked spike S_{11} is marked in red. The differences in the computing formulations between different methods are enclosed by boxes of various colors.

A Mutual Information

Mutual information (MI), first formalized in Shannon's information theory [51], provides a generalized dependence measure between random variables. Unlike measures limited to linear relationships, MI captures complex statistical dependencies through probability distribution analysis. This property makes it suitable for quantifying similarity between discrete spike features in SNNs. The MI between two spike features $\boldsymbol{S}$ and $\boldsymbol{S}'$ is expressed as:

$$I(\boldsymbol{S}; \boldsymbol{S}') = \sum_{s \in \boldsymbol{S}} \sum_{s' \in \boldsymbol{S}'} p(s, s') \log \left(\frac{p(s, s')}{p(s)p(s')} \right) \tag{10}$$

where s and s' indicate the values in feature S and S', respectively. $p(s, s')$ is the joint probability distribution, and $p(s)$ and $p(s')$ are the marginal probability distributions. MI satisfies two fundamental properties: (1) Non-negativity ($I(\boldsymbol{S}; \boldsymbol{S}') \geq 0$) ensures meaningful information measures, and a higher MI value indicates greater similarity. $I(\boldsymbol{S}; \boldsymbol{S}') = 0$, if and only if $\boldsymbol{S}$ and $\boldsymbol{S}'$ are independent random variables. (2) Symmetry ($I(\boldsymbol{S}; \boldsymbol{S}') = I(\boldsymbol{S}'; \boldsymbol{S})$) reflects bidirectional dependence.

B Analysis of MI-TRQR

B.1 Analysis of Forward Propagation

SNNs become a promising alternative to ANNs, contributing to their advantages in energy efficiency. Unlike ANNs, SNNs achieve inter-layer communication with binary signals (0-nothing or 1-spike) [36]. As a kind of neuromorphic computing algorithm, SNNs only compute sparse spike-based accumulation (AC) and avoid handling the zero-value inputs [48], such as the calculation of membrane potential $\boldsymbol{U} = \boldsymbol{W}\boldsymbol{S}$, as shown in Fig. 5. The energy consumption of SNNs E_{Conv} is influenced by the spiking firing rates of input features. Referring to the equation in [45], the inference energy consumption of a network is computed as follows:

$$E_{net} = E_{MAC} \cdot FL_{Conv}^{1} + E_{AC} \cdot T \cdot (\sum_{n=2}^{N} FL_{Conv}^{n} \cdot fr_{Conv}^{n} + \sum_{m=1}^{M} FL_{FC}^{m} \cdot fr_{FC}^{m}) \tag{11}$$

where FL_{Conv}^{n}, FL_{FC}^{m}, fr_{Conv}^{n}, and fr_{FC}^{m} are the FLOPs and spiking firing rate of the n-th convolution and m-th FC layer. N and M are the total number of convolution and FC layers. E_{MAC} and E_{AC} are the energy consumption of MAC and AC operation. Following the works [74, 19], we assume that $E_{MAC} = 4.6pJ$ and $E_{AC} = 0.9pJ$.

In forward propagation, our MI-TRQR removes some spikes. We can see the effect on the calculation of membrane potential $\boldsymbol{U}_{22}^{l+1}$:

$$\text{Vanilla: } \boldsymbol{U}_{22}^{l+1} = \boldsymbol{W}_{11} + \boldsymbol{W}_{13} + \boldsymbol{W}_{32} \tag{12}$$

$$\text{MI-TRQR: } \boldsymbol{U}_{22}^{l+1} = \boldsymbol{W}_{13} + \boldsymbol{W}_{32} \tag{13}$$

where l indicates the layer number. We can see that MI-TRQR requires less accumulation than the vanilla method.

B.2 Analysis of Backward Propagation

In the backward propagation, we discuss the partial derivative of loss on different variables with different methods in the following three cases. Firstly, we denote the partial derivative of loss on the $l+1$-th layer membrane potential $\boldsymbol{U}^{l+1}$ as:

$$\boldsymbol{\delta} = \frac{\partial L}{\partial \boldsymbol{U}^{l+1}} \tag{14}$$

where $\boldsymbol{\delta}$ is the partial derivative, as shown in Fig. 5.

Case 1: The partial derivative of loss on weight. Similar to ANNs, the partial derivative is:

$$\frac{\partial L}{\partial \boldsymbol{W}_{m,n}} = \sum_{i=1}^{H} \sum_{j=1}^{W} \boldsymbol{\delta}_{i,j} \boldsymbol{S}_{i+m-2,j+n-2} \tag{15}$$

where H and W indicate the height and width of features. The difference is that the spike feature $\boldsymbol{S}$ is a sparse binary tensor, meaning the computing of Eq. 15 is spike-based accumulation. For example, with different methods, the partial derivative of loss on $\boldsymbol{W}_{22}$ is:

$$\text{Vanilla: } \frac{\partial L}{\boldsymbol{W}_{22}} = \boldsymbol{\delta}_{11} + \boldsymbol{\delta}_{13} + \boldsymbol{\delta}_{32} + \boldsymbol{\delta}_{44} \tag{16}$$

$$\text{MI-TRQR: } \frac{\partial L}{\boldsymbol{W}_{22}} = \boldsymbol{\delta}_{13} + \boldsymbol{\delta}_{32} + \boldsymbol{\delta}_{44} \tag{17}$$

We can observe that the gradient calculation of MI-TRQR requires less accumulation than the vanilla method, meaning the parameter update in MI-TRQR removes the redundant/invalid gradient in the vanilla method.

Case 2: The partial derivative of loss on spike features is computed with:

$$\frac{\partial L}{\partial \boldsymbol{S}_{m,n}} = \sum_{i=1}^{k} \sum_{j=1}^{k} \boldsymbol{W}_{i,j} \boldsymbol{\delta}_{m-i+2,n-j+2} \tag{18}$$

where k indicates the size of the convolution kernel. However, the calculation of this partial derivative has a precondition: The input feature needs to involve the forward calculation, which means that the input feature must be non-zero. In other words, the gradient of the zero input is zero. Thus, the partial derivative of loss on the spike $\boldsymbol{S}_{11}$ is:

$$\text{Vanilla: } \frac{\partial L}{\boldsymbol{S}_{11}} = \boldsymbol{W}_{11}\boldsymbol{\delta}_{22} + \boldsymbol{W}_{12}\boldsymbol{\delta}_{21} + \boldsymbol{W}_{21}\boldsymbol{\delta}_{12} + \boldsymbol{W}_{22}\boldsymbol{\delta}_{11} \tag{19}$$

$$\text{MI-TRQR: } \frac{\partial L}{\boldsymbol{S}_{11}} = 0 \tag{20}$$

We can see that the partial derivative of loss on the removed spike in the two methods is completely different. Thus, we perceive that removing a spike means removing its derivative/gradient.

Case 3: The partial derivative of loss on the l-th layer membrane potential $\boldsymbol{U}^l$ is easily obtained based on case 2. This partial derivative can be represented as:

$$\frac{\partial \boldsymbol{L}}{\partial \boldsymbol{U}^l} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{S}}\frac{\partial \boldsymbol{S}}{\partial \boldsymbol{U}^l} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{S}}\text{Hea}'(\boldsymbol{U}^l) \tag{21}$$

where $\text{Hea}'(\boldsymbol{U}^l)$ indicates the derivative of $\text{Hea}(\boldsymbol{U}^l)$. Specifically, with different methods, the partial derivative of loss on the membrane potential $\boldsymbol{U}^l_{11}$ is:

$$\text{Vanilla: } \frac{\partial L}{\boldsymbol{U}^l_{11}} = \frac{\partial \boldsymbol{L}}{\partial \boldsymbol{S}_{11}}\text{Hea}'(\boldsymbol{U}^l_{11}) \tag{22}$$

$$\text{MI-TRQR: } \frac{\partial L}{\boldsymbol{U}^l_{11}} = 0 \times \text{Hea}'(\boldsymbol{U}^l_{11}) = 0 \tag{23}$$

We can see that the removed spike can influence the gradient of the membrane potential at the previous layer.

Summary. Through the above formula derivation and analysis, we can conclude that removing the spikes after the last convolutional layer can affect the derivative of the previous membrane potential, and further affect the weight gradient and parameter updates at the previous layers. In this way, MI-TRQR can learn compact and powerful representations in SNNs.

C Experiments

C.1 Experimental Setup

Datasets.

CIFAR10/100 [26], two small datasets, contain 50,000 training and 10,000 test samples. ImageNet [6] is a large-scale dataset of about 1.3 million images (1.25 million for training and 0.5 million for testing) across 1,000 classes. CIFAR10-DVS [31], an event-stream dataset converted from CIFAR10 with dynamic vision sensors, includes 10,000 event streams in 10 classes. DVS128 Gait [59] contains 4200 samples in 20 classes from 21 volunteers. Electricity [28] captures hourly electricity consumption measured in kilowatt-hours (kWh).

Implementation. We adopt the PSN [14] and STMixer [7] as baselines of the classification task. We adopt the CPG-PE [35] as the baseline of the time-series forecasting task. We follow their experimental setup, such as network architecture, training methods, data preprocessing, etc. All experiments were conducted on a Ubuntu 20.04.6 LTS server with 8 NVIDIA GeForce RTX 3090. The MI dependence is calculated with the TorchMetrics package.

For DVS128 Gait, we designed a small 3B-Net. Its structure is c128k3s1-BN-PLIF-SEW Block-MPk2s2*3-FC20. Here, c128k3s1 denotes a convolution layer with channels 128, kernel size 3, and stride 1. PLIF refers to the PLIF spiking neuron [13]. MPk2s2 represents the max pooling with kernel size 2 and stride 2. SEW Block is a custom-designed SEW block, expressed as c32k3s1-BN-PLIF-c128k3s1-BN-PLIF. The symbol *3 indicates that the structure in is repeated three times.

Table 7: Comparison with other methods on CIFAR10/100.

Dataset	Method	Backbone	T	Accuracy (↑%)	Power (↓mJ)
CIFAR10	TAB [24]	ResNet19	4	94.76	-
	RevSFormer [80]	RevSFormer-4-384	4	95.34	-
	TCJA-SNN [88]	ResNet18	4	95.60	-
	PSN [14]	Modified PLIF Net	4	95.32	1.32
	MI-TRQR (Ours)	Modified PLIF Net	4	**95.83**±0.10	**0.35**±0.05
	STATA [90]	Spikingformer	4	95.8	-
	STAtten [29]	SDT-2-512	4	96.03	-
	STMixer [7]	STMixer-4-384-32	4	96.01	0.95
	MI-TRQR (Ours)	STMixer-4-384-32	4	**96.64**±0.12	**0.51**±0.08
CIFAR100	SLT-TET [1]	ResNet19	4	75.01	-
	NDOT$_A$ [23]	VGG11	4	76.18	-
	LietE-SNN [32]	-	6	77.10	-
	PSN$^{\spadesuit}$ [14]	ResNet18	4	75.75	0.43
		ResNet19	4	76.14	1.78
	MI-TRQR (Ours)	ResNet18	4	**76.70**±0.12	**0.31**±0.08
		ResNet19	4	**77.70**±0.10	**1.24**±0.07
	ST [16]	ST-4-384	4	79.69	-
	SNN-ViT [58]	SDT	4	80.1	-
	SDT+SEMM [85]	SDT	4	80.26	-
	STMixer [7]	STMixer-4-384-32	4	81.87	1.08
	MI-TRQR (Ours)	STMixer-4-384-32	4	**83.06**±0.11	**0.77**±0.08

Table 8: Impact of *local redundancy* (LR) and *global redundancy* (GR) on CIFAR10.

Method	LR	GR	T	Accuracy (↑%)	Firing rate (↓%)
PSN [14]	-	-	4	95.32	16.39
MI-TRQR (ours)	✓	-	4	95.53 (↑0.21)	10.59 (↓35.39)
	-	✓	4	95.77 (↑0.45)	8.98 (↓45.21)
	✓	✓	4	**95.83** (↑0.51)	**5.36** (↓67.30)

C.2 Experimental Results on CIFAR10/100

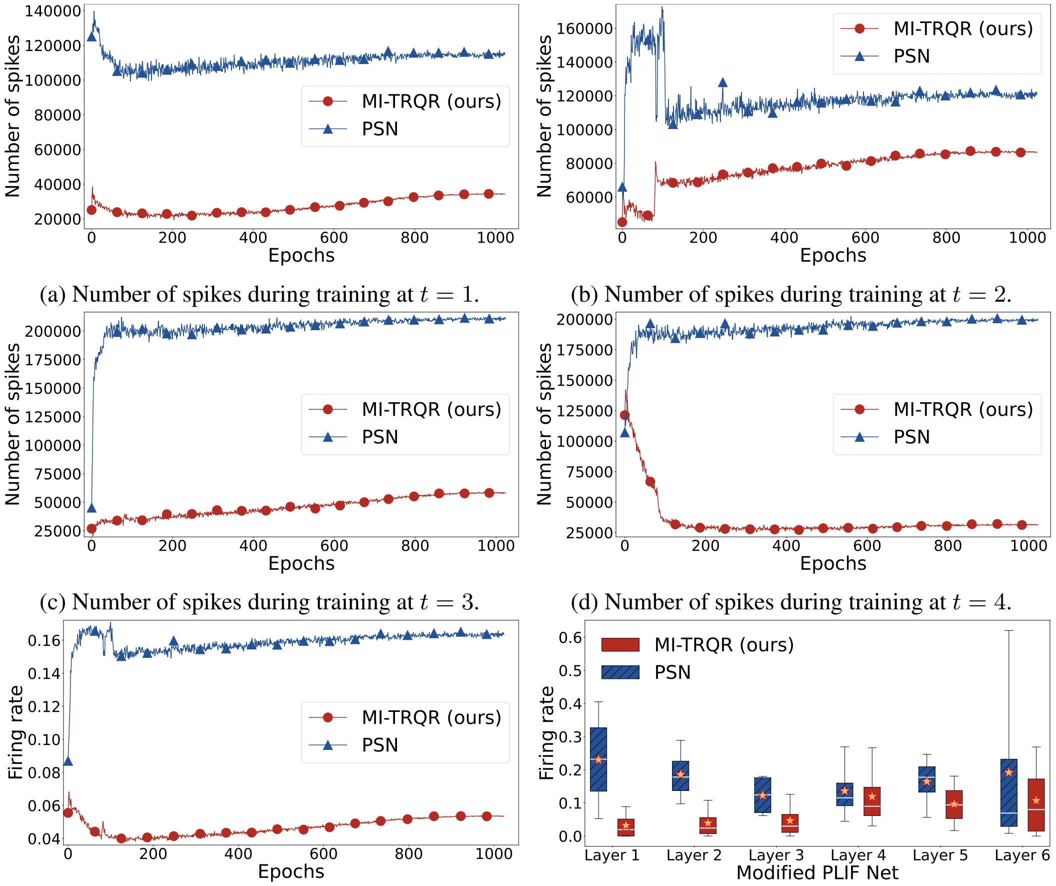
C.2.1 Comparison with other methods

CIFAR10. Tab. 7 presents the classification performance and energy consumption of various methods on CIFAR-10. MI-TRQR achieves **95.83%** accuracy, outperforming PSN (95.32%) by +0.51% while reducing energy consumption by **73.48%** (**0.35mJ** vs. 1.32mJ). With the STMixer-4-384-32 backbone, MI-TRQR attains **96.64%** accuracy, surpassing STMixer (96.01%) by 0.63% while reducing energy consumption by **46.3%** (**0.51 mJ** vs. 0.95 mJ).

CIFAR100. As shown in Tab. 7, MI-TRQR consistently outperforms PSN across both ResNet18 and ResNet19 backbones, achieving **76.70%** and **77.70%** accuracy respectively, compared to 75.75% and 76.14% of PSN, while also reducing power consumption from 0.43 mJ and 1.78 mJ to **0.31 mJ** and **1.24 mJ**. With the STMixer-4-384-32 backbone, MI-TRQR attains **83.06%** accuracy, outperforming the original STMixer (81.87%) by **1.19%** while reducing energy consumption by **28.70%** (**0.77 mJ** vs. 1.08 mJ).

C.2.2 Ablation Study on CIFAI10

We conduct ablation studies on CIFAR10 to validate the effectiveness of our redundancy designs in MI-TRQR. As summarized in Tab. 8, both local and global redundancy contribute to performance



(a) Number of spikes during training at $t = 1$.

(b) Number of spikes during training at $t = 2$.

(c) Number of spikes during training at $t = 3$.

(d) Number of spikes during training at $t = 4$.

(e) Network spiking firing rate during training.

(f) Layer-wise spiking firing rate of the network.

Figure 6: Comparison of the number of spikes and firing rate in different methods on CIFAR10.

improvements over the PSN [14]. Using local redundancy alone improves accuracy by 0.21% while reducing the firing rate by 35.39%. Feature redundancy demonstrates stronger effects, achieving a 0.45% accuracy gain with 45.21% fewer spikes. Notably, combining both mechanisms yields the best results - our full model attains 95.83% accuracy (+0.51%) with only 5.36% firing rate, representing a 67.30% reduction in spike activity. The number of spikes and firing rates in various methods are shown in Fig. 6. Specifically, the number of spikes at four timesteps during training is shown from Fig. 6a to 6d. It can be seen that our MI-TRQR consistently fired fewer spikes than PSN at each timestep. The overall spiking firing rates during training are plotted in Fig. 6e. The firing rate of our MI-TRQR is reduced by 67.29% compared to that of PSN (5.36% vs. 16.39%). The layer-wise firing rates of the Modified PLIF Net are presented in Fig. 6f. Our MI-TRQR has a lower firing rate than PSN at every layer of the Modified PLIF Net.

C.3 Extended Ablation Studies

C.3.1 Spike normalization

To verify the effect of spike normalization in Eq. 5, we apply identical removal ratios to all spikes. Specifically, after obtaining the mask $\boldsymbol{M}_t$ with $p_{t,,h,w}$, we shuffle its elements to generate a random mask for spike removal. Results in Tab. 9 indicate that both the MI-based ratio and the removed spikes play essential roles.

In Tab. 10, we evaluate different static penalty factors that increase with timesteps and serve as a regularizer on the number of spikes. The results show that these static penalties effectively suppress redundant spikes and improve overall performance. Notably, MI-TRQR achieves higher accuracy with a lower firing rate.

Table 9: Effect of a random mask on CIFAR10-DVS when T=4.

Method	Shuffle	T	Accuracy(↑%)	Firing rate(↓%)
PSN [14]	-	4	82.3	14.47
MI-TRQR (ours)	✓	4	83.4 (↑1.1)	8.51 (↓41.19)
	-	4	**84.0** (↑1.7)	**8.35** (↓42.29)

Table 10: Effect of static penalty factors on CIFAR10-DVS when T=4.

Method	$t=1$	$t=2$	$t=3$	$t=4$	Accuracy(↑%)	Firing rate(↓%)
PSN [14]	-	-	-	-	82.3	14.47
Static penalty	0	0.1	0.2	0.3	83.0 (↑0.7)	10.83 (↓25.16)
	0	0.15	0.3	0.45	82.9 (↑0.6)	9.55 (↓34.00)
	0	0.2	0.4	0.6	82.8 (↑0.5)	10.61 (↓26.68)
MI-TRQR (ours)	-	-	-	-	**84.0** (↑1.7)	**8.35** (↓42.29)

Table 11: Effect of different techniques on CIFAR10-DVS when T=4.

Method	Technique	T	Accuracy(↑%)	Firing rate(↓%)
PSN [14]	-	4	82.3	14.47
Absolute distance	Euclidean	4	82.4 (↑0.1)	11.74 (↓18.87)
Linear similarity	Pearson	4	82.5 (↑0.2)	10.92 (↓24.53)
Feature direction	Cosine	4	82.9 (↑0.6)	9.59 (↓33.72)
MI-TRQR (ours)	MI	4	**84.0** (↑1.7)	**8.35** (↓42.29)

Table 12: Effect of different factors on CIFAR10-DVS when T=4.

Method	Factor a	T	Accuracy(↑%)	Firing rate(↓%)
PSN [14]	-	4	82.3	14.47
MI-TRQR (ours)	0.5	4	84.5 (↑2.2)	10.66 (↓26.33)
	1	4	84.0 (↑1.7)	8.35 (↓42.29)
	2	4	82.7 (↑0.4)	8.16 (↓43.61)

C.3.2 Other techniques

We replace MI with simpler similarity measures, including Euclidean similarity (via 1/(1+Euclidean distance)), Pearson correlation, and Cosine similarity. As shown in Tab. 11, spiking firing rates still decrease with these alternatives, while MI-TRQR achieves higher accuracy and a lower firing rate.

C.3.3 Scaling factor

We multiply $p_{t,h,w}$ (Eq. 5) with an additional factor a, and evaluate the results under different setting in Tab. 12. As shown, the factor a influences the trade-off between accuracy and firing rate.

C.3.4 Mask generation strategies

In our method, we adopt Bernoulli-based masks to remove redundant spikes in a data-dependent manner, without introducing additional hyperparameters. This design also admits a clear theoretical interpretation. To further validate its effectiveness, we compare it with a global redundancy-based thresholding approach for pixel-wise spike removal. Specifically, for the feature $S_t \in \{0,1\}^{C\times H\times W}(t > 1)$, we compute the global redundancy p_t^g (Eq. 3) and the local redundancy $p_{t,h,w}^l$ (Eq. 4). If $p_{t,h,w}^l > p_t^g$, the spike feature $S_{t,h,w} \in \{0,1\}^C$ is set into 0. As shown in Tab. 13, MI-TRQR with Bernoulli-based masks achieves better performance.

Table 13: Effect of mask generation strategies on CIFAR10-DVS when T=4.

Method	Mask type	T	Accuracy(↑%)	Firing rate(↓%)
PSN [14]	-	4	82.3	14.47
MI-TRQR (ours)	Global threshold	4	82.6 (↑0.3)	8.97 (↓38.01)
	Bernoulli	4	**84.0** (↑1.7)	**8.35** (↓42.29)

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state our contributions in the field of SNNs.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in the final section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The paper provides a complete proof of the proposed method in Appendix B.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We will upload our code and data to Github upon acceptance. We have shown our experiment results in the Experiment Section, which can be reproduced by referring to the submitted code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The dataset used in this article is publicly available, and the source code will be uploaded to ensure that others can reproduce the experimental results.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have shown our experimental settings and implementation details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report the mean as well as the standard deviation accuracy in experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The energy consumption is provided in the experiment section.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read and confirmed that the research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The original paper for datasets we used are all cited.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We adopt public datasets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: The paper does not involve LLMs as any important, original, or non-standard components.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.