TRIM: TEMPORAL-SUBSPACE RECONSTRUCTION INDUCED BY MOMENTS FOR POST-TRAINING STRUCTURED PRUNING OF SPIKING NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking neural networks (SNNs) offer energy-efficient computation through sparse and event-driven processing, but their growing scale imposes substantial storage and computational burdens. Existing structured pruning methods for SNNs typically rely on sparsity-aware training or post-pruning fine-tuning, making model compression expensive. To our knowledge, we propose the first post-training structured pruning framework for SNNs that requires neither retraining nor fine-tuning. We begin by establishing a surrogate membrane-potential reconstruction baseline that uses a small calibration set to guide structure selection and weight compensation. However, under tight structural budgets and given the distinctive dynamics of SNNs, accurately reconstructing full temporal trajectories would require the retained structures to capture both low-order temporal components and fine-grained temporal variations. To address this challenge, we propose TRIM, namely Temporal-subspace Reconstruction Induced by Moments, which projects surrogate membrane potential trajectories onto a fixed temporal subspace that captures their mean level and linear temporal trend. This shifts the reconstruction objective toward preserving these low-order temporal components rather than matching entire trajectories. Experiments across diverse visual SNNs and spiking language models demonstrate competitive accuracy–compression trade-offs without any fine-tuning.

1 INTRODUCTION

Spiking neural networks (SNNs) offer a biologically inspired, energy-efficient computing paradigm in which neurons communicate through discrete spikes rather than the real-valued activations of conventional artificial neural networks (ANNs) (Maass, 1997; Roy et al., 2019). Neurons in SNNs integrate incoming signals into their membrane potentials over time, enabling history-dependent spatiotemporal processing (Gerstner et al., 2014). With binary spikes, synaptic computation uses accumulations (ACs) rather than multiply–accumulate (MAC) operations, while sparse firing allows event-driven implementations to skip updates when spikes are absent. Neuromorphic platforms such as Loihi (Davies et al., 2018) and Speck (Yao et al., 2024b) exploit these properties for low-power neural processing. Alongside these hardware developments, SNN architectures have been applied to image classification (Zhou et al., 2023b), object detection (Li et al., 2025), and language processing (Xing et al., 2024), extending spike-based computation in various learning tasks.

Recent advances in SNN architectures (Yao et al., 2024a; Lee et al., 2025) and training strategies (Yan et al., 2025) have improved task performance. However, deep SNNs can retain substantial parameter redundancy and associated storage and computational costs (Li et al., 2024; Shen et al., 2025). Sparse firing can reduce synaptic computation, but does not inherently reduce connectivity or weight storage (Hua et al., 2026). This motivates compressing pretrained SNNs while preserving task performance.

Among existing approaches to model compression, pruning has been widely adopted to remove redundant parameters or computational units (Liu et al., 2019; Cheng et al., 2024). Pruning methods are broadly categorized as unstructured or structured according to the granularity of removed units (Cheng et al., 2024). Unstructured pruning eliminates individual weights or connections, allow-

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

ing flexible parameter selection without changing the original tensor dimensions (Han et al., 2015; Louizos et al., 2018). However, the resulting irregular sparsity is often difficult to translate into wall-clock speedups on general-purpose hardware without specialized sparse support (Wen et al., 2016; Ji et al., 2018). Structured pruning instead removes complete groups, such as filters, channels, neurons, attention heads, or layers, thereby producing smaller dense operators (Li et al., 2017a; He et al., 2017; Fang et al., 2023). Although structured pruning offers less flexibility and must account for structural dependencies across connected layers (Chen et al., 2022; Fang et al., 2023), the resulting regular networks can be more easily executed using standard libraries and accelerators, making structured pruning attractive for practical model compression (Wen et al., 2016; Li et al., 2017a).

Given the memory and computational constraints of practical deployment, pruning offers a promising approach to improving SNN efficiency. Structured SNN pruning methods, including SCA (Li et al., 2024) and QP-SNN (Wei et al., 2025), produce hardware-friendly models by removing channels or kernels, but rely on sparse training or post-pruning fine-tuning. These training requirements become increasingly burdensome as spiking models scale, particularly for large language models (Xing et al., 2025). Preserving accuracy under structured pruning without retraining or fine-tuning therefore remains an important challenge. In this paper, we introduce a post-training structured pruning framework for resource-constrained deployment of spiking models, requiring neither retraining nor fine-tuning. Specifically, we first establish a surrogate membrane potential reconstruction baseline that couples structure selection with weight compensation while accounting for temporal accumulation. However, reconstructing complete temporal trajectories becomes increasingly difficult as fewer structures are retained, which can degrade accuracy at high pruning ratios. To address this limitation, we introduce TRIM, which performs reconstruction within a fixed low-order temporal subspace. We summarize our main contributions as follows:

- We establish a surrogate membrane potential reconstruction baseline for post-training structured pruning of SNNs, combining channel selection, closed-form weight compensation, and asymmetric calibration.
- To improve the baseline at high pruning ratios, we introduce TRIM, which uses two orthogonal temporal moments to prioritize low-order temporal components of the surrogate membrane potential over complete trajectory matching.
- Experiments on visual SNNs and spiking language models demonstrate competitive performance–compression trade-offs, with TRIM improving upon the surrogate membrane potential reconstruction baseline in most evaluated settings.

2 Related Work

SNN pruning techniques. SNN pruning includes unstructured, structured, and semi-structured approaches. Unstructured methods remove individual synapses through activity-dependent rules (Rathi et al., 2019), sparsity-inducing optimization (Yin et al., 2021), or joint weight–activity sparsification (Shi et al., 2024; Chen et al., 2025). Structured methods remove channels using principal components of membrane potentials (Chowdhury et al., 2021), spiking channel activity (Li et al., 2024), or spike-activity singular values (Wei et al., 2025); recent work explores operation-constrained structure search (Xie et al., 2025) and energy-aware gating (She et al., 2026). Semi-structured methods impose $N : M$ sparsity patterns (Ye et al., 2025). Beyond pruning granularity, sparse training adapts connectivity through rewiring (Bellec et al., 2018), compressibility-aware structure learning (Shen et al., 2025), and topology-guided regrowth (Hua et al., 2026). Many approaches still require training or fine-tuning, increasing the cost of compressing pretrained SNNs.

Post-training structured pruning. Post-training structured pruning compresses pretrained networks using limited calibration data without retraining or fine-tuning. In vision models, representative methods use sensitivity-based sampling (Liebenwein et al., 2020), relevance-based filter ranking (Yeom et al., 2021), or causal interventions (Bauvin et al., 2025). Language-model approaches include Fisher-based structural selection with output reconstruction (Kwon et al., 2022), activation-fluctuation scoring with bias compensation in FLAP (An et al., 2024), and second-order weight updates in SlimGPT (Ling et al., 2024). Complementary methods remove low-variance representation dimensions in SliceGPT (Ashkboos et al., 2024) or group neurons by function in FANG (Yu et al., 2026). However, these ANN-oriented formulations do not explicitly account for membrane integration and spike generation, limiting their direct applicability to SNNs.

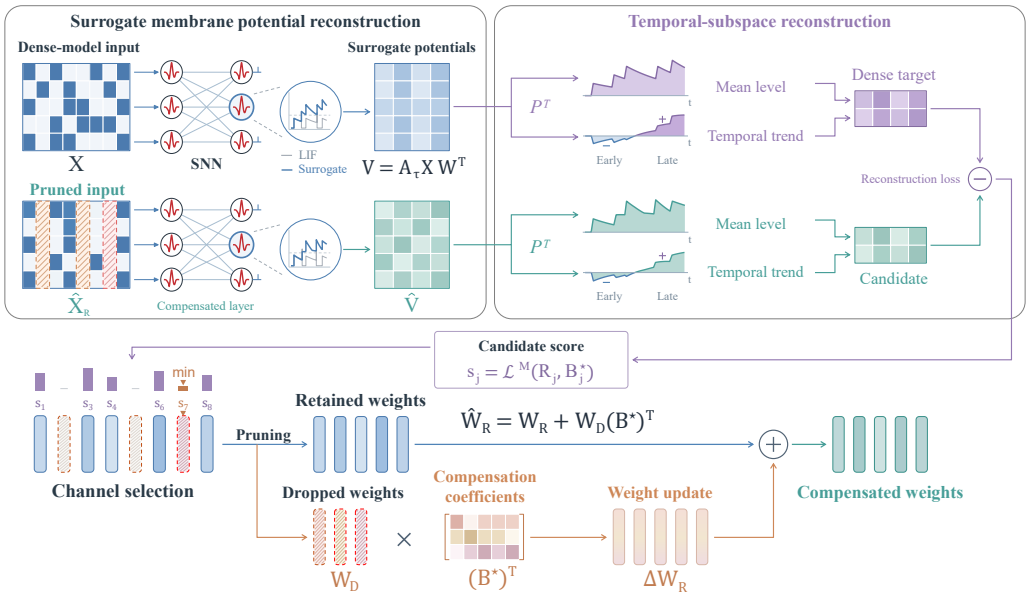


Figure 1: Overview of the TRIM framework. Surrogate membrane potentials preserve leaky integration; two temporal moments capture their mean level and linear trend. TRIM removes the channel with the lowest compensated reconstruction loss against the fixed dense target. Closed-form combinations of dropped weights update the retained weights.

Other SNN efficiency techniques. Beyond pruning, the efficiency of SNNs can also be improved through quantization, knowledge distillation, and optimization of temporal inference. Quantization reduces parameter storage through low-precision weights (Qiu et al., 2025), while knowledge distillation transfers teacher representations and predictions to improve the accuracy of low-latency SNNs (Xu et al., 2025; Yu et al., 2025). Additionally, confidence-based early termination and flexible time-step training enable adjustment of inference duration to balance accuracy and computational cost (Li et al., 2023; Du et al., 2025). These techniques offer complementary avenues for improving efficiency alongside pruning, and they are not discussed in detail here.

3 Method

In this section, we first establish a surrogate membrane potential reconstruction baseline for post-training structured SNN pruning, coupling channel selection and weight compensation. We then introduce TRIM based on reconstruction in a fixed temporal subspace. Figure 1 illustrates its pipeline.

3.1 Preliminaries

Neuron model. Spiking-neuron models vary in biological detail (Hodgkin & Huxley, 1952; Zhang et al., 2022; Wei et al., 2023); we use the computationally efficient leaky integrate-and-fire (LIF) model (Wu et al., 2018). At time step t, the pre-reset membrane potential in layer ℓ is

$$\widetilde{\mathbf{U}}^{\ell}[t] = \tau \mathbf{U}^{\ell}[t-1] + \mathbf{I}^{\ell}[t], \tag{1}$$

where $\widetilde{\mathbf{U}}^{\ell}[t]$ and $\mathbf{U}^{\ell}[t]$ are the pre- and post-reset potentials, $\tau \in (0,1)$ is the decay factor, and $\mathbf{I}^{\ell}[t] = \mathbf{W}^{\ell}\mathbf{S}^{\ell-1}[t]$ is the synaptic current from the weights $\mathbf{W}^{\ell}$ and input spikes $\mathbf{S}^{\ell-1}[t]$. Neuron j spikes when its pre-reset potential reaches the threshold θ:

$$S_j^{\ell}[t] = \begin{cases} 1, & \widetilde{U}_j^{\ell}[t] \geq \theta, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

With $\odot$ denoting element-wise multiplication, the hard-reset update is

$$\mathbf{U}^{\ell}[t] = \widetilde{\mathbf{U}}^{\ell}[t] \odot \big(\mathbf{1} - \mathbf{S}^{\ell}[t]\big). \tag{3}$$

Structured pruning. We consider a linear layer with synaptic weights $\mathbf{W}^\ell \in \mathbb{R}^{C_{\text{out}} \times C_{\text{in}}}$ and input spikes $\mathbf{S}^{\ell-1}[t] \in \{0,1\}^{C_{\text{in}}}$, where C_{in} and C_{out} denote the input and output channel counts. Channel pruning removes columns of $\mathbf{W}^\ell$ together with the corresponding upstream output channels, subject to architectural dependencies (Fang et al., 2023). For convolutional layers, all kernel coefficients associated with an input channel are removed jointly (Meng et al., 2024). Given a retained width K, we select a retained channel set $\mathcal{R} \subseteq \{1, \ldots, C_{\text{in}}\}$ with $|\mathcal{R}| = K$, and denote the dropped channels by $\mathcal{D} = \{1, \ldots, C_{\text{in}}\} \setminus \mathcal{R}$. The pruning ratio is $\rho = 1 - K/C_{\text{in}}$, and the channel selection remains fixed across samples and time steps. With $\mathbf{W}^\ell_{\mathcal{R}} = \mathbf{W}^\ell[:, \mathcal{R}]$ and $\mathbf{S}^{\ell-1}_{\mathcal{R}}[t] = \mathbf{S}^{\ell-1}[t]_{\mathcal{R}}$, and analogous definitions for $\mathcal{D}$, the original synaptic current can be written as

$$\mathbf{I}^\ell[t] = \mathbf{W}^\ell \mathbf{S}^{\ell-1}[t] = \underbrace{\mathbf{W}^\ell_{\mathcal{R}} \mathbf{S}^{\ell-1}_{\mathcal{R}}[t]}_{\text{retained}} + \underbrace{\mathbf{W}^\ell_{\mathcal{D}} \mathbf{S}^{\ell-1}_{\mathcal{D}}[t]}_{\text{dropped}}. \tag{4}$$

Holding the layer input fixed, dropping the channels in $\mathcal{D}$ without weight compensation gives

$$\mathbf{I}^\ell_{\text{prune}}[t] = \mathbf{W}^\ell_{\mathcal{R}} \mathbf{S}^{\ell-1}_{\mathcal{R}}[t], \qquad \mathbf{W}^\ell_{\mathcal{R}} \in \mathbb{R}^{C_{\text{out}} \times K}. \tag{5}$$

The contribution $\mathbf{W}^\ell_{\mathcal{D}} \mathbf{S}^{\ell-1}_{\mathcal{D}}[t]$ of the dropped channels is therefore absent from the pruned synaptic current. To compensate for such pruning errors, we need to further adjust the retained weights using calibration data.

3.2 Surrogate Membrane Potential Reconstruction

Surrogate membrane potential. Due to neuronal dynamics in Eq. (1), perturbations caused by channel removal can affect subsequent membrane states. To retain this temporal accumulation in a tractable reconstruction objective, we adopt a surrogate membrane potential that preserves leaky integration. Given a calibration sample q, let $\mathbf{X}^\ell_q = [\mathbf{S}^{\ell-1}_q[1], \ldots, \mathbf{S}^{\ell-1}_q[T]]^\top \in \mathbb{R}^{T \times C_{\text{in}}}$ collect the input spikes of the unpruned network over T time steps. With zero initial state and decay factor τ, the temporal accumulation is represented by a lower-triangular matrix $\mathbf{A}_\tau \in \mathbb{R}^{T \times T}$ with entries as,

$$[\mathbf{A}_\tau]_{ts} = \begin{cases} \tau^{t-s}, & s \le t, \\ 0, & s > t, \end{cases} \qquad t, s \in \{1, \ldots, T\}. \tag{6}$$

The surrogate membrane-potential trajectory is then given by

$$\mathbf{V}^\ell_q = \mathbf{A}_\tau \mathbf{X}^\ell_q (\mathbf{W}^\ell)^\top \in \mathbb{R}^{T \times C_{\text{out}}}. \tag{7}$$

Here, $\mathbf{A}_\tau \mathbf{X}^\ell_q$ denotes the leak-integrated input. This surrogate is used only for reconstruction, i.e. the network's firing and reset rules remain unchanged.

Asymmetric calibration. When pruning successive layers, the current layer receives input spikes already affected by the preceding pruning steps. We therefore use the surrogate membrane-potential trajectory of the unpruned layer as a fixed reconstruction target for the pruned layer. For each calibration sample q, $\mathbf{X}^\ell_q$ comes from the frozen unpruned network, whereas $\widehat{\mathbf{X}}^\ell_q \in \mathbb{R}^{T \times C_{\text{in}}}$ comes from the progressively pruned network before channel selection in the current layer. For retained channels $\mathcal{R}$, define $\widehat{\mathbf{X}}^\ell_{q,\mathcal{R}} = \widehat{\mathbf{X}}^\ell_q[:, \mathcal{R}]$ and let $\widehat{\mathbf{W}}^\ell_{\mathcal{R}} \in \mathbb{R}^{C_{\text{out}} \times K}$ denote the compensated weights. The reconstruction loss over N calibration samples is

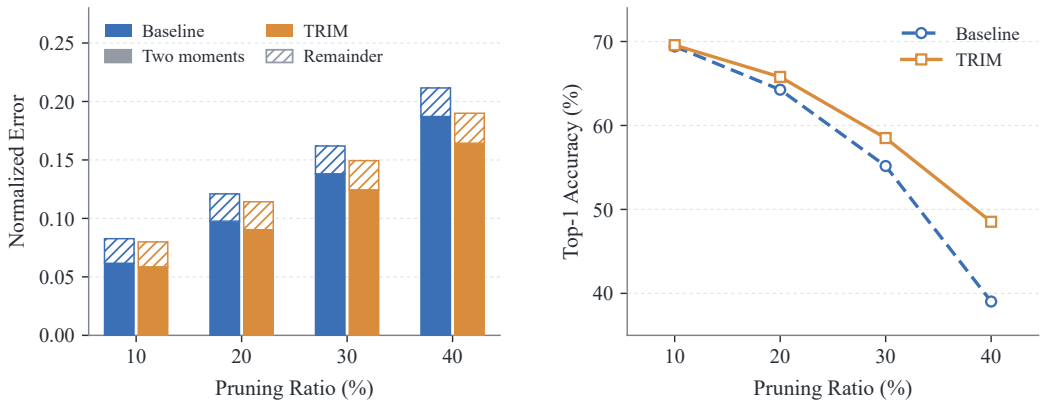
$$\mathcal{L}_\ell(\mathcal{R}, \widehat{\mathbf{W}}^\ell_{\mathcal{R}}) = \frac{1}{NT} \sum_{q=1}^{N} \left\| \mathbf{A}_\tau \left(\mathbf{X}^\ell_q (\mathbf{W}^\ell)^\top - \widehat{\mathbf{X}}^\ell_{q,\mathcal{R}} (\widehat{\mathbf{W}}^\ell_{\mathcal{R}})^\top \right) \right\|_F^2. \tag{8}$$

Channel selection and weight compensation. Channel selection should account for how well the retained channels can compensate for a deletion. Motivated by evidence that representational directions can be distributed across neurons (Raghu et al., 2017), we approximate the leak-integrated inputs of dropped channels as linear combinations of retained-channel inputs:

$$\mathbf{A}_\tau \widehat{\mathbf{X}}^\ell_{q,\mathcal{D}} \approx \mathbf{A}_\tau \widehat{\mathbf{X}}^\ell_{q,\mathcal{R}} \mathbf{B}^\ell, \tag{9}$$

where $\mathbf{B}^\ell \in \mathbb{R}^{|\mathcal{R}| \times |\mathcal{D}|}$ contains linear-combination coefficients shared across calibration samples and time steps. This approximation motivates the following compensated weights:

$$\widehat{\mathbf{W}}^\ell_{\mathcal{R}} = \mathbf{W}^\ell_{\mathcal{R}} + \mathbf{W}^\ell_{\mathcal{D}} (\mathbf{B}^\ell)^\top. \tag{10}$$



(a) Reconstruction error decomposition

(b) Classification accuracy

Figure 2: Comparison of the surrogate membrane potential reconstruction baseline and TRIM on VGG16/CIFAR-100. (a) The two-moment component (solid) and the orthogonal remainder (hatched) sum to the full-trajectory reconstruction error. Both are normalized by the squared Frobenius norm of the unpruned target. (b) Top-1 accuracy at different pruning ratios.

Substituting Eq. (10) into Eq. (8), the reconstruction objective can be expressed as,

$$\mathcal{L}_\ell(\mathcal{R}, \mathbf{B}^\ell) = \frac{1}{NT} \sum_{q=1}^{N} \left\| \mathbf{A}_\tau \left[(\mathbf{X}_q^\ell - \widehat{\mathbf{X}}_q^\ell)(\mathbf{W}^\ell)^\top + (\widehat{\mathbf{X}}_{q,\mathcal{D}}^\ell - \widehat{\mathbf{X}}_{q,\mathcal{R}}^\ell \mathbf{B}^\ell)(\mathbf{W}_{\mathcal{D}}^\ell)^\top \right] \right\|_F^2 . \tag{11}$$

The two terms inside the brackets capture inherited input discrepancies and the residual after current-layer pruning and compensation, respectively. We denote the full residual inside the norm by $\mathbf{E}_q^\ell$ and seek to minimize the loss over $\mathcal{R}$ and $\mathbf{B}^\ell$, subject to $|\mathcal{R}| = K$. Let $\mathbf{H}_0$ and $\mathbf{C}$ denote the normalized current-input Gram and current-to-dense cross-Gram matrices of the leak-integrated inputs. We use the diagonally loaded matrices $\mathbf{H} = \mathbf{H}_0 + \lambda_x \mathbf{I}$ and $\mathbf{G} = (\mathbf{W}^\ell)^\top \mathbf{W}^\ell + \lambda_w \mathbf{I}$, with layer-fixed $\lambda_x, \lambda_w > 0$. For fixed $\mathcal{R}$, minimizing the diagonally loaded objective $\mathcal{L}_\ell$ over $\mathbf{B}^\ell$ yields

$$\mathbf{B}_{\mathcal{R}}^{\ell\star} = \mathbf{H}_{\mathcal{R}\mathcal{R}}^{-1} \left(\mathbf{C}_{\mathcal{R},:} \mathbf{G}_{:,\mathcal{D}} - \mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{G}_{\mathcal{R}\mathcal{D}} \right) \mathbf{G}_{\mathcal{D}\mathcal{D}}^{-1}. \tag{12}$$

Here, $\star$ denotes the minimizer; subscripts select rows and columns, with ":" selecting all indices.

Starting with all channels retained, we remove the channel whose deletion minimizes the damped objective after optimal compensation:

$$j^\star \in \arg\min_{j \in \mathcal{R}} \mathcal{L}_\ell \left(\mathcal{R} \setminus \{j\}, \mathbf{B}_{\mathcal{R}\setminus\{j\}}^{\ell\star} \right). \tag{13}$$

We repeat this deletion until K channels remain, reusing layer-wise calibration statistics and scoring candidates via rank-one updates without separate reconstruction solves. Using Eq. (10), the final compensation can be absorbed into the retained weights, so pruning requires neither retraining nor additional inference modules. The full derivation is given in the Appendix.

3.3 TRIM: Temporal-Subspace Reconstruction

Problem analysis. The baseline seeks to reconstruct the full surrogate membrane potential trajectory. However, under a limited channel budget, it is difficult for the retained channels and their shared compensation coefficients to faithfully reproduce both low-order temporal components and finer temporal variations. Therefore, it is necessary to determine which components dominate the reconstruction error as pruning becomes more aggressive.

To characterize each T-step trajectory, we use a fixed temporal basis $\mathbf{P} = [\mathbf{p}_0, \mathbf{p}_1] \in \mathbb{R}^{T\times 2}$, comprising constant and centered linear basis vectors, with $\bar{t} = (T+1)/2$:

$$[\mathbf{p}_0]_t = \frac{1}{\sqrt{T}}, \qquad [\mathbf{p}_1]_t = \frac{t - \bar{t}}{\sqrt{\sum_{s=1}^{T} (s - \bar{t})^2}}. \tag{14}$$

Projecting the trajectory onto $\mathbf{p}_0$ and $\mathbf{p}_1$ gives two temporal moments, capturing the mean level over time and the linear trend from early to late steps, respectively. Since $\mathbf{P}^\top\mathbf{P} = \mathbf{I}_2$, the residual in Eq. (11) decomposes as:

$$\|\mathbf{E}_q^\ell\|_F^2 = \underbrace{\|\mathbf{P}^\top\mathbf{E}_q^\ell\|_F^2}_{\text{two-moment component}} + \underbrace{\|(\mathbf{I}_T - \mathbf{P}\mathbf{P}^\top)\mathbf{E}_q^\ell\|_F^2}_{\text{orthogonal remainder}}. \tag{15}$$

Figure 2(a) shows the decomposition of the normalized reconstruction error for VGG16 on CIFAR-100. As the pruning ratio increases from 10% to 40%, the two-moment component accounts for approximately 97.3% of the increase in the baseline's total reconstruction error. This observation motivates prioritizing the reduction of the two-moment reconstruction error under a limited channel budget.

Temporal-subspace reconstruction. We represent the dense and progressively pruned inputs as $\mathbf{Z}_q^\ell = \mathbf{P}^\top\mathbf{A}_\tau\mathbf{X}_q^\ell$ and $\widehat{\mathbf{Z}}_q^\ell = \mathbf{P}^\top\mathbf{A}_\tau\widehat{\mathbf{X}}_q^\ell$, respectively. Both representations belong to $\mathbb{R}^{2\times C_{\text{in}}}$. With the compensated weights parameterized as in Eq. (10), we define the temporal-subspace reconstruction objective as

$$\mathcal{L}_\ell^{\text{M}}(\mathcal{R}, \mathbf{B}^\ell) = \frac{1}{2N}\sum_{q=1}^{N}\left\|\mathbf{Z}_q^\ell(\mathbf{W}^\ell)^\top - \widehat{\mathbf{Z}}_{q,\mathcal{R}}^\ell(\widehat{\mathbf{W}}_{\mathcal{R}}^\ell)^\top\right\|_F^2. \tag{16}$$

We optimize this objective under the same constraint $|\mathcal{R}| = K$. Since the residual is precisely $\mathbf{P}^\top\mathbf{E}_q^\ell$, the objective targets the two-moment error component in Eq. (15).

Moment projection preserves the quadratic structure of the objective, allowing us to reuse the baseline's reconstruction procedure. We apply Eqs. (12)–(13) using calibration statistics computed from the moment representations, with the same weight metric and damping scheme.

In Figure 2(a), TRIM's reduction in the two-moment error outweighs the increase in the orthogonal remainder, yielding lower total reconstruction error at all evaluated pruning ratios. Figure 2(b) shows that TRIM also achieves higher classification accuracy than the baseline at all four pruning ratios. TRIM combines hardware-friendly channel pruning with a competitive accuracy–compression trade-off, providing a practical solution for deploying pretrained spiking models in resource-constrained settings without retraining or fine-tuning.

4 Experiments

4.1 Experimental Setup

Models and compared methods. Our visual experiments cover VGG16, ResNet20, Spikingformer-4-384, SEW-ResNet18, and VGGSNN. For language understanding, we evaluate SpikingLM (Liang et al., 2026) and SpikeLM enhanced with SmoothSpike, denoted as SpikeLM + SmoothSpike (Xing et al., 2024; Zhou et al., 2026). We compare against magnitude pruning (MP) and Wanda-sp, the structured variant of Wanda (Sun et al., 2024; An et al., 2024), in both domains. Visual comparisons additionally include QP-SNN (Wei et al., 2025), OBC (Frantar & Alistarh, 2022), and FLAP (An et al., 2024). These visual baselines are adapted to the same structural pruning interfaces; QP-SNN uses its pruning criterion alone, without quantization. We also include the baseline from Section 3.2 to assess the contribution of temporal-subspace reconstruction. All pruned models are evaluated without retraining or fine-tuning.

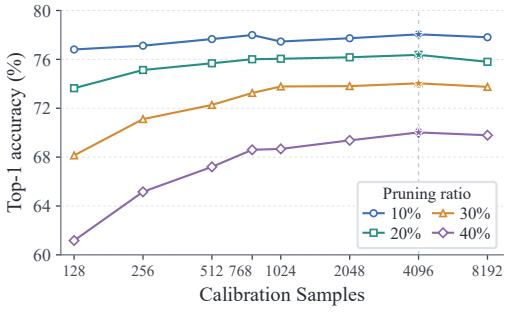


Figure 3: Accuracy versus calibration-set size for TRIM on ResNet20/CIFAR-100.

Datasets and evaluation. We evaluate visual classification on CIFAR-10 and CIFAR-100 (Krizhevsky, 2009), TinyImageNet (Le & Yang, 2015), ImageNet-1K (Deng et al., 2009), and the

Table 1: Top-1 accuracy (%) of pruned visual SNNs without retraining or fine-tuning. Parentheses indicate unpruned accuracy; PR denotes the pruning ratio. Best and second-best results in each row are bold and underlined, respectively. Baseline denotes the reconstruction method in Section 3.2. † QP-SNN uses its pruning criterion only, without quantization.

Dataset	Model (Acc.)	PR (%)	MP	Wanda-sp	QP-SNN†	OBC	FLAP	Baseline	TRIM
CIFAR-10	VGG16 (93.63%)	10	84.01	71.84	77.14	92.51	90.32	92.66	**93.12**
		20	50.58	34.87	32.93	90.85	84.31	91.01	**91.35**
		30	27.95	24.59	19.19	88.55	62.17	87.60	**88.92**
		40	13.78	13.27	17.04	82.00	62.17	81.13	**84.05**
		50	17.89	10.09	10.22	69.52	9.99	68.52	**74.31**
	ResNet20 (96.36%)	10	94.03	90.40	86.19	95.90	95.53	95.87	**95.99**
		20	84.68	72.52	50.30	95.48	92.78	95.51	**95.61**
		30	46.23	44.87	24.95	94.91	85.54	95.07	**95.15**
		40	23.45	23.50	23.16	93.75	65.63	93.85	**94.01**
		50	16.49	20.33	17.32	**92.18**	26.47	91.92	91.77
CIFAR-100	VGG16 (71.52%)	10	45.66	41.68	41.56	69.10	62.44	69.41	**69.57**
		20	11.40	11.10	10.98	64.87	42.81	64.26	**65.78**
		30	4.25	2.61	3.18	58.45	12.67	55.17	**58.50**
		40	1.73	1.21	1.08	45.80	1.00	39.03	**48.53**
	ResNet20 (79.49%)	10	71.33	59.62	57.86	**78.36**	75.09	77.73	78.05
		20	48.88	24.88	26.72	75.91	66.41	76.20	**76.37**
		30	25.47	6.40	7.00	73.91	50.67	73.67	**74.04**
		40	5.20	1.58	1.70	69.88	25.32	69.88	**70.02**
		50	1.99	1.10	1.31	60.68	6.55	63.15	**64.11**
	Spikingformer (79.09%)	10	61.17	74.06	71.59	76.86	76.20	76.56	**77.88**
		20	18.65	54.28	51.28	73.39	64.83	**73.88**	73.19
		30	4.47	36.61	31.22	68.76	47.20	**68.99**	68.00
TinyImageNet	VGG16 (59.08%)	10	21.68	14.32	14.18	53.93	37.03	55.17	**55.54**
		20	5.03	4.85	3.81	44.95	15.22	47.50	**49.92**
		30	2.17	1.93	1.71	30.65	3.09	34.81	**41.38**
ImageNet-1K	SEW-ResNet18 (63.27%)	10	36.71	37.55	37.54	**60.16**	54.23	59.24	59.81
		20	10.24	10.70	11.09	54.95	40.95	54.58	**55.45**
		30	2.31	5.50	3.67	48.03	26.81	48.50	**49.76**
		40	0.94	1.11	1.22	36.79	14.57	40.46	**42.21**
		50	0.65	0.57	0.65	23.66	5.55	29.20	**31.76**
DVS-CIFAR10	VGGSNN (82.10%)	10	77.80	72.40	58.60	80.10	77.30	80.20	**80.80**
		20	64.20	59.50	33.50	80.20	63.80	80.20	**80.40**
		30	37.10	44.40	29.70	77.80	32.40	77.90	**80.00**
		40	24.20	30.40	18.30	75.40	12.20	77.80	**78.30**

neuromorphic dataset DVS-CIFAR10 (Li et al., 2017b). Language evaluation covers eight GLUE tasks: MNLI, QQP, QNLI, SST-2, CoLA, STS-B, MRPC and RTE (Wang et al., 2019). We report the top-1 accuracy for visual classification and the corresponding accuracy, F1, or correlation scores for language tasks.

Calibration-set size. We determine calibration-set sizes through preliminary experiments on visual SNNs. Figure 3 presents ResNet20 on CIFAR-100 as a representative example, using 128–8192 calibration samples at pruning ratios of 10%–40%. Accuracy generally improves with more calibration data, but the largest tested calibration set provides no further improvement. These observations inform the calibration-set selection before the main visual comparisons.

4.2 Performance and Efficiency on Visual SNNs

Across the eight model–dataset pairs in Table 1, TRIM outperforms MP, Wanda-sp, the pruning-only QP-SNN variant and FLAP in all 34 evaluated settings, achieving the highest accuracy in 29. Compared to OBC, TRIM achieves 41.38% accuracy with VGG16 on TinyImageNet at 30% pruning and

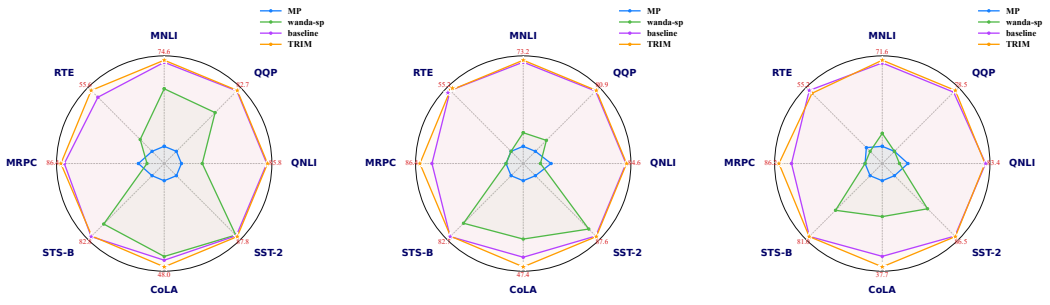
31.76% with SEW-ResNet18 on ImageNet-1K at 50% pruning, corresponding to improvements of 10.73 and 8.10 percentage points, respectively. TRIM also improves upon the baseline in 31 settings. For VGG16 on CIFAR-100, it raises accuracy from 39.03% to 48.53% at 40% pruning, supporting the benefit of temporal-subspace reconstruction under a limited channel budget. Table 2 compares compression efficiency against the unpruned models, using 30% pruning for VGG16 and 40% for ResNet20 and VGGSNN. On DVS-CIFAR10, VGGSNN retains 78.30% accuracy, compared with 82.10% before pruning, while reducing model size, SOPs, and estimated inference energy by approximately 57.6%, 62.2%, and 59.8%, respectively. These savings are achieved while retaining 32-bit weights, demonstrating that TRIM reduces storage and computational requirements through structural pruning alone. Together, these results support TRIM as an effective approach to post-training compression of visual SNNs without retraining or fine-tuning.

Table 2: Efficiency metrics comparison of TRIM with full-precision uncompressed SNNs.

	Architecture	Connection	Bit	Model size (MB)	SOPs (M)	Power (mJ)	Accuracy
CIFAR-10	VGG16	100%	32	58.88	165.93	0.182	93.63%
	VGG16	52.30%	32	30.80	82.85	0.098	88.92%
CIFAR-100	ResNet20	100%	32	68.36	797.79	0.735	79.49%
	ResNet20	60.13%	32	41.11	471.51	0.441	70.02%
DVS-CIFAR10	VGGSNN	100%	32	37.06	1153.11	1.162	82.10%
	VGGSNN	42.42%	32	15.73	436.00	0.467	78.30%

4.3 Performance on Spiking Language Models

Table 3 and Figure 4 report GLUE performance for pruned SpikingLM and SpikeLM + SmoothSpike, respectively. On SpikingLM, TRIM achieves the highest mean score among the compared methods at all four evaluated pruning ratios. At 70% pruning, it retains a mean score of 72.2, compared with 77.2 for the unpruned model, outperforming MP and Wanda-sp by 29.0 and 19.5 points, respectively. Beyond these comparisons, TRIM improves upon the baseline by 0.6–0.8 points in mean score across the four pruning ratios, supporting the benefit of temporal-subspace reconstruction over full-trajectory reconstruction for preserving language-task performance. Figure 4 shows similar benefits on SpikeLM + SmoothSpike, where TRIM outperforms the baseline in most evaluated task–pruning-ratio combinations. For example, at 50% pruning, TRIM increases the CoLA score from 42.1 to 47.4. Together, these results support TRIM's effectiveness in preserving task performance across visual SNNs and spiking language models under post-training structured pruning without retraining or fine-tuning.



(a) 30% pruning (b) 50% pruning (c) 70% pruning

Figure 4: TRIM vs. competing methods on eight GLUE tasks with SpikeLM + SmoothSpike. (a) 30%, (b) 50%, and (c) 70% pruning.

Table 3: GLUE performance of pruned SpikingLM without retraining or fine-tuning. Best and second-best scores for each metric at each pruning ratio are bold and underlined, respectively.

Method	PR (%)	MNLI-m/mm	QQP_{F1}	QNLI	SST-2	CoLA	STS-B	MRPC_{F1}	RTE	MEAN
SpikingLM	0%	75.7/75.9	84.5	84.6	90.1	50.4	82.5	87.9	61.7	77.2
MP	10%	69.5/69.0	78.5	74.5	86.2	16.4	68.3	84.6	57.4	66.9
Wanda-sp	10%	**74.5**/75.0	82.8	83.9	88.3	43.4	81.3	86.0	59.2	74.9
Baseline	10%	**74.5**/74.6	**83.8**	**84.3**	88.5	43.1	82.7	86.7	59.9	75.4
TRIM	10%	74.4/**75.1**	**83.8**	84.1	**88.8**	**46.7**	**82.8**	**87.0**	**60.3**	**76.0**
MP	30%	53.3/53.8	54.5	62.7	68.5	0.0	41.1	81.7	55.2	52.1
Wanda-sp	30%	66.8/67.0	72.6	67.7	87.8	40.4	73.4	81.2	56.0	68.3
Baseline	30%	73.0/74.0	82.7	83.4	88.1	42.4	**81.8**	86.0	58.8	74.5
TRIM	30%	**73.5/74.2**	**82.9**	**83.6**	**88.4**	**45.9**	81.4	**86.2**	**59.2**	**75.1**
MP	50%	45.9/46.8	53.8	59.0	55.0	0.0	15.0	81.2	55.2	45.6
Wanda-sp	50%	50.1/51.5	58.7	55.7	83.9	30.7	67.1	81.2	55.2	60.3
Baseline	50%	71.6/72.4	**81.1**	82.3	87.8	40.3	**81.7**	85.5	**58.8**	73.6
TRIM	50%	**72.2/72.8**	80.8	**82.4**	**88.2**	**45.3**	81.3	**86.2**	58.5	**74.4**
MP	70%	39.6/39.3	53.8	58.0	50.9	0.0	9.6	81.2	52.3	43.2
Wanda-sp	70%	44.3/46.3	53.8	55.5	70.5	15.0	49.6	81.2	51.6	52.7
Baseline	70%	69.6/70.5	78.0	**81.3**	86.8	31.7	79.8	85.3	**58.5**	71.4
TRIM	70%	**70.6/70.8**	**78.7**	81.1	**87.0**	**36.1**	**80.0**	**86.0**	58.1	**72.2**

Table 4: Size of the pruned model and time required for pruning (SpikingLM-110M).

Pruning ratio	# of parameters	Memory size	Time required for pruning a layer
0% (FP32)	109.51M	417.75 MiB	–
10%	101.49M	387.14 MiB	3.87 s
20%	93.46M	356.52 MiB	6.56 s
30%	85.41M	325.83 MiB	8.60 s

Table 4 summarizes the parameter count, FP32 storage, and pruning time per layer of SpikingLM at different pruning ratios. Both parameter count and storage decrease monotonically as the pruning ratio increases, confirming the storage savings of structured pruning. In contrast, pruning time increases with more aggressive pruning. This trend is consistent with TRIM's sequential channel-selection procedure: for a layer with C_{in} input channels and K retained channels, selection requires $C_{\text{in}} - K$ successive deletions, with a cumulative rank-one update cost of $O((C_{\text{in}} - K)C_{\text{in}}^2)$. Reusing layer-wise calibration statistics avoids repeated data processing, but does not eliminate the additional selection and compensation updates required at higher pruning ratios. Overall, TRIM offers a practical approach to post-training compression with a manageable pruning cost in the evaluated settings. These results motivate further exploration of its applicability to larger spiking language models.

5 Conclusion

In this paper, we present a post-training structured pruning framework for SNNs without retraining or fine-tuning. We couple channel selection with weight compensation by scoring each candidate deletion using the surrogate membrane-potential reconstruction loss after compensation. We further introduce TRIM, which prioritizes low-order temporal components rather than complete trajectory matching. Experiments on visual SNNs and spiking language models demonstrate competitive performance–compression trade-offs, with improvements over the surrogate membrane potential reconstruction baseline in most evaluated settings. Moreover, compensation is incorporated directly into the retained weights, requiring no additional inference modules. In general, our findings highlight the value of jointly considering structure selection and reconstruction-objective design, supporting TRIM as a practical approach to compressing pretrained spiking models for resource-constrained deployment.

LLM USE DISCLOSURE

We used large language models to (i) improve the grammar and wording of this manuscript and (ii) assist with experimentation by drafting portions of the code and suggesting debugging steps. All scientific ideas, claims, experiment designs, and final implementations are by the authors. We reviewed and verified all LLM-assisted text and code before use, and the authors remain fully responsible for the content of this paper.

ETHICS STATEMENT

This work complies with the ICLR Code of Ethics (`https://iclr.cc/public/CodeOfEthics`). Our study does not involve human subjects, sensitive personal data, or any form of biometric information. All datasets used are publicly available and widely adopted in the research community. We have taken care to avoid generating or reinforcing harmful content, stereotypes, or biases in both model design and evaluation. No proprietary or confidential data was used. There are no known conflicts of interest or external sponsorship that could influence the outcomes of this research. We acknowledge the importance of ethical considerations in AI research and have made efforts to ensure transparency, reproducibility, and fairness throughout the development of our framework.

REPRODUCIBILITY STATEMENT

The proposed method is described in Section 3, with detailed derivations provided in Appendix A. The models, datasets, and evaluation settings are specified in Section 4.1. Training configurations and data preprocessing details for the unpruned visual SNNs are provided in Appendix C. Our code will be made publicly available.

REFERENCES

Yongqi An, Xu Zhao, Tao Yu, Ming Tang, and Jinqiao Wang. Fluctuation-based adaptive structured pruning for large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(10):10865–10873, 2024.

Saleh Ashkboos, Maximilian Croci, Marcelo Gennari do Nascimento, Torsten Hoefler, and James Hensman. SliceGPT: Compress large language models by deleting rows and columns. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 11682–11701, 2024.

Baptiste Bauvin, Loïc Baret, and Ola Ahmad. C-SWAP: Explainability-aware structured pruning for efficient neural networks compression. In *36th British Machine Vision Conference 2025, BMVC 2025, Sheffield, UK, November 24-27, 2025*. BMVA, 2025.

Guillaume Bellec, David Kappel, Wolfgang Maass, and Robert Legenstein. Deep rewiring: Training very sparse deep networks. In *International Conference on Learning Representations*, 2018.

Jue Chen, Huan Yuan, Jianchao Tan, Bin Chen, Chengru Song, and Di Zhang. Resource constrained model compression via minimax optimization for spiking neural networks. In *Proceedings of the 31st ACM International Conference on Multimedia*, 2023. doi: 10.1145/3581783.3611838.

Shuo Chen, Zeshi Liu, and Haihang You. Maximizing energy efficiency in spiking neural networks: A dynamic joint pruning framework. In *2025 62nd ACM/IEEE Design Automation Conference (DAC)*, pp. 1–7, 2025.

Tianlong Chen, Xuxi Chen, Xiaolong Ma, Yanzhi Wang, and Zhangyang Wang. Coarsening the granularity: Towards structurally sparse lottery tickets. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 3025–3039. PMLR, 17–23 Jul 2022.

Hongrong Cheng, Miao Zhang, and Javen Qinfeng Shi. A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10558–10578, 2024.

Sayeed Shafayet Chowdhury, Isha Garg, and Kaushik Roy. Spatio-temporal pruning and quantization for low-latency spiking neural networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9. IEEE, 2021.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE, 2009.

Kangrui Du, Yuhang Wu, Shikuang Deng, and Shi Gu. Temporal flexibility in spiking neural networks: Towards generalization across time steps and deployment friendliness. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 59917–59936, 2025.

Gongfan Fang, Xinyin Ma, Mingli Song, Michael Bi Mi, and Xinchao Wang. DepGraph: Towards any structural pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16091–16101, June 2023.

Elias Frantar and Dan Alistarh. Optimal brain compression: A framework for accurate post-training quantization and pruning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 4475–4488. Curran Associates, Inc., 2022.

Wulfram Gerstner, Werner M. Kistler, Richard Naud, and Liam Paninski. *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press, 2014.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28, pp. 1135–1143. Curran Associates, Inc., 2015.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, October 2017.

A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544, 1952.

Mark Horowitz. Computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pp. 10–14. IEEE, 2014. doi: 10.1109/ISSCC.2014.6757323.

Yuan Hua, Jilin Zhang, Yingtao Zhang, Leyi You, Baobo Xiong, Carlo Vittorio Cannistraci, and Hong Chen. Cannistraci-hebb training on ultra-sparse spiking neural networks. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 114755–114779, 2026.

Yu Ji, Ling Liang, Lei Deng, Youyang Zhang, Youhui Zhang, and Yuan Xie. TETRIS: TilE-matching the TRemendous irregular sparsity. In *Advances in Neural Information Processing Systems*, volume 31, pp. 4115–4125. Curran Associates, Inc., 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Woosuk Kwon, Sehoon Kim, Michael Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. A fast post-training pruning framework for transformers. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 24101–24116. Curran Associates, Inc., 2022.

Ya Le and Xuan Yang. Tiny ImageNet visual recognition challenge. Technical report, Stanford University, 2015.

Donghyun Lee, Yuhang Li, Youngeun Kim, Shiting Xiao, and Priyadarshini Panda. Spiking transformer with spatial-temporal attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13948–13958, June 2025.

Chen Li, Edward G Jones, and Steve Furber. Unleashing the potential of spiking neural networks with dynamic confidence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13350–13360, October 2023.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient ConvNets. In *International Conference on Learning Representations*, 2017a.

Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. CIFAR10-DVS: An event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:309, 2017b.

Yaxin Li, Qi Xu, Jiangrong Shen, Hongming Xu, Long Chen, and Gang Pan. Towards efficient deep spiking neural networks construction with spiking activity based pruning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 29063–29073. PMLR, 21–27 Jul 2024.

Ziqi Li, Tao Gao, Yisheng An, Ting Chen, Jing Zhang, Yuanbo Wen, Mengkun Liu, and Qianxi Zhang. Brain-inspired spiking neural networks for energy-efficient object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3552–3562, June 2025.

Yu Liang, Zijian Zhou, Wenjie Wei, Shuai Wang, Honglin Cao, Ammar Belatreche, Yu Yang, Malu Zhang, Yang Yang, and Haizhou Li. SpikingLM: Towards fully spiking language model. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Lucas Liebenwein, Cenk Baykal, Harry Lang, Dan Feldman, and Daniela Rus. Provable filter pruning for efficient neural networks. In *International Conference on Learning Representations*, 2020.

Gui Ling, Ziyang Wang, Yuliang Yan, and Qingwen Liu. SlimGPT: Layer-wise structured pruning for large language models. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 107112–107137. Curran Associates, Inc., 2024.

Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019.

Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through L_0 regularization. In *International Conference on Learning Representations*, 2018.

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

Xiang Meng, Shibal Ibrahim, Kayhan Behdin, Hussein Hazimeh, Natalia Ponomareva, and Rahul Mazumder. OSSCAR: One-shot structured pruning in vision and language models with combinatorial optimization. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 35354–35377. PMLR, 21–27 Jul 2024.

Xuerui Qiu, Malu Zhang, Jieyuan Zhang, Wenjie Wei, Honglin Cao, Junsheng Guo, Rui-Jie Zhu, Yimeng Shan, Yang Yang, and Haizhou Li. Quantized spike-driven transformer. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 14477–14501, 2025.

Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. SVCCA: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30, pp. 6076–6085. Curran Associates, Inc., 2017.

Nitin Rathi, Priyadarshini Panda, and Kaushik Roy. STDP-based pruning of connections and weight quantization in spiking neural networks for energy-efficient recognition. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 38(4):668–677, 2019.

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

Yongzhi She, Qihua Zhou, Yuhao Wang, Yaodong Huang, Laizhong Cui, and Jingcai Guo. Spik4lite: Refactoring neuromorphic sparsity for efficient spiking neural networks on commodity edge devices. In *International Conference on Machine Learning*, 2026.

Jiangrong Shen, Qi Xu, Gang Pan, and Badong Chen. Improving the sparse structure learning of spiking neural networks from the view of compression efficiency. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 92834–92846, 2025.

Xinyu Shi, Jianhao Ding, Zecheng Hao, and Zhaofei Yu. Towards energy efficient spiking neural networks: An unstructured pruning framework. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 50818–50838, 2024.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 4942–4964, 2024.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019.

Wenjie Wei, Malu Zhang, Hong Qu, Ammar Belatreche, Jian Zhang, and Hong Chen. Temporal-coded spiking neural networks with dynamic firing threshold: Learning with event-driven backpropagation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10552–10562, October 2023.

Wenjie Wei, Malu Zhang, Zijian Zhou, Ammar Belatreche, Yimeng Shan, Yu Liang, Honglin Cao, Jieyuan Zhang, and Yang Yang. QP-SNN: Quantized and pruned spiking neural networks. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 96470–96495, 2025.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems*, volume 29, pp. 2074–2082. Curran Associates, Inc., 2016.

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

Hui Xie, Yuhe Liu, Shaoqi Yang, Jinyang Guo, Yufei Guo, Yuqing Ma, Jiaxin Chen, Jiaheng Liu, and Xianglong Liu. SPEAR: Structured pruning for spiking neural networks via synaptic operation estimation and reinforcement learning. *arXiv preprint arXiv:2507.02945*, 2025.

Xingrun Xing, Zheng Zhang, Ziyi Ni, Shitao Xiao, Yiming Ju, Siqi Fan, Yequan Wang, Jiajun Zhang, and Guoqi Li. SpikeLM: Towards general spike-driven language modeling via elastic bi-spiking mechanisms. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 54698–54714. PMLR, 21–27 Jul 2024.

Xingrun Xing, Boyan Gao, Zheng Liu, David Clifton, Shitao Xiao, Wanpeng Zhang, Li Du, Zheng Zhang, Guoqi Li, and Jiajun Zhang. SpikeLLM: Scaling up spiking neural network to large language models via saliency-based spiking. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 32880–32899, 2025.

Zekai Xu, Kang You, Qinghai Guo, Xiang Wang, and Zhezhi He. BKDSNN: Enhancing the performance of learning-based spiking neural networks training with blurred knowledge distillation. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol (eds.), *Computer Vision – ECCV 2024*, pp. 106–123, Cham, 2025. Springer Nature Switzerland.

Jiaqi Yan, Changping Wang, De Ma, Huajin Tang, Qian Zheng, and Gang Pan. Training high performance spiking neural network by temporal model calibration. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 70289–70308. PMLR, 13–19 Jul 2025.

Man Yao, Jiakui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo XU, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 52885–52907, 2024a.

Man Yao, Ole Richter, Guangshe Zhao, Ning Qiao, Yannan Xing, Dingheng Wang, Tianxiang Hu, Wei Fang, Tugba Demirci, Michele De Marchi, Lei Deng, Tianyi Yan, Carsten Nielsen, Sadique Sheik, Chenxi Wu, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15:4464, 2024b.

Shuhan Ye, Yi Yu, Qixin Zhang, Chenqi Kong, Qiangqiang Wu, Xudong Jiang, and Dacheng Tao. Sparse by rule: Probability-based N:M pruning for spiking neural networks. *arXiv preprint arXiv:2511.12097*, 2025.

Seul-Ki Yeom, Philipp Seegerer, Sebastian Lapuschkin, Alexander Binder, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Pruning by explaining: A novel criterion for deep neural network pruning. *Pattern Recognition*, 115:107899, 2021.

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*, pp. 2017–2025. ACM, 2021.

Kairong Yu, Chengting Yu, Tianqing Zhang, Xiaochen Zhao, Shu Yang, Hongwei Wang, Qiang Zhang, and Qi Xu. Temporal separation with entropy regularization for knowledge distillation in spiking neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8806–8816, June 2025.

Tao Yu, Yongqi An, Kuan Zhu, Guibo Zhu, Ming Tang, and Jinqiao Wang. Improving generalization in LLM structured pruning via function-aware neuron grouping. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(33):27969–27977, 2026.

Malu Zhang, Jiadong Wang, Jibin Wu, Ammar Belatreche, Burin Amornpaisannon, Zhixuan Zhang, V. P. K. Miriyala, Hong Qu, Yansong Chua, Trevor E. Carlson, and Haizhou Li. Rectified linear postsynaptic potential function for backpropagation in deep spiking neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(5):1947–1958, 2022.

Chenlin Zhou, Liutao Yu, Zhaokun Zhou, Han Zhang, Zhengyu Ma, Huihui Zhou, and Yonghong Tian. Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. *arXiv preprint arXiv:2304.11954*, 2023a. URL https://arxiv.org/abs/2304.11954v1.

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

Zijian Zhou, Wenjie Wei, Yu Liang, Jialin Li, Ammar Belatreche, Honglin Cao, Shuai Wang, Malu Zhang, Yang Yang, and Haizhou Li. SmoothSpike: Spiking transformer with learnable hadamard transformation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

A Detailed Derivations

A.1 Quadratic Objective and Closed-Form Solution

We omit the layer index ℓ for clarity. Let $c = C_{\text{in}}$, with $\mathbf{W} \in \mathbb{R}^{C_{\text{out}} \times c}$ and $\mathbf{X}_q, \widehat{\mathbf{X}}_q \in \mathbb{R}^{T \times c}$ denoting the dense and progressively pruned inputs, respectively.

Substituting Eq. (10) into the residual in Eq. (8) gives

$$\begin{aligned}\mathbf{E}_q &= \mathbf{A}_\tau \left[\mathbf{X}_q \mathbf{W}^\top - \widehat{\mathbf{X}}_{q,\mathcal{R}} \left(\mathbf{W}_{\mathcal{R}}^\top + \mathbf{B}\mathbf{W}_{\mathcal{D}}^\top\right)\right] \\ &= \mathbf{A}_\tau \left[(\mathbf{X}_q - \widehat{\mathbf{X}}_q)\mathbf{W}^\top + \widehat{\mathbf{X}}_{q,\mathcal{D}}\mathbf{W}_{\mathcal{D}}^\top - \widehat{\mathbf{X}}_{q,\mathcal{R}}\mathbf{B}\mathbf{W}_{\mathcal{D}}^\top\right] \\ &= \mathbf{A}_\tau \left[(\mathbf{X}_q - \widehat{\mathbf{X}}_q)\mathbf{W}^\top + \left(\widehat{\mathbf{X}}_{q,\mathcal{D}} - \widehat{\mathbf{X}}_{q,\mathcal{R}}\mathbf{B}\right)\mathbf{W}_{\mathcal{D}}^\top\right].\end{aligned}$$

The second equality uses $\widehat{\mathbf{X}}_q \mathbf{W}^\top = \widehat{\mathbf{X}}_{q,\mathcal{R}} \mathbf{W}_{\mathcal{R}}^\top + \widehat{\mathbf{X}}_{q,\mathcal{D}} \mathbf{W}_{\mathcal{D}}^\top$. Summing $\|\mathbf{E}_q\|_F^2$ over q and dividing by NT gives Eq. (11).

The calibration statistics used in Section 3.2 are

$$\mathbf{H}_0 = \frac{1}{NT} \sum_{q=1}^{N} \widehat{\mathbf{X}}_q^\top \mathbf{A}_\tau^\top \mathbf{A}_\tau \widehat{\mathbf{X}}_q, \qquad \mathbf{C} = \frac{1}{NT} \sum_{q=1}^{N} \widehat{\mathbf{X}}_q^\top \mathbf{A}_\tau^\top \mathbf{A}_\tau \mathbf{X}_q. \tag{17}$$

Both matrices belong to $\mathbb{R}^{c \times c}$. Here, $\mathbf{H}_0$ is the current-input Gram matrix, whereas $\mathbf{C}$ is the current-to-dense cross-Gram matrix and need not be symmetric.

Expanding Eq. (11) gives a quadratic in $\mathbf{B}$. Replacing $\mathbf{H}_0$ by $\mathbf{H} = \mathbf{H}_0 + \lambda_x \mathbf{I}$ and $\mathbf{W}^\top \mathbf{W}$ by $\mathbf{G} = \mathbf{W}^\top \mathbf{W} + \lambda_w \mathbf{I}$ throughout this expansion, while leaving $\mathbf{C}$ unchanged, gives the damped objective

$$\begin{aligned}\mathcal{L}(\mathcal{R}, \mathbf{B}) = \mathcal{L}(\mathcal{R}, \mathbf{0}) &+ \operatorname{tr}\left(\mathbf{G}_{\mathcal{D}\mathcal{D}} \mathbf{B}^\top \mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{B}\right) \\ &- 2 \operatorname{tr}\left[\mathbf{B}^\top \left(\mathbf{C}_{\mathcal{R},:} \mathbf{G}_{:,\mathcal{D}} - \mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{G}_{\mathcal{R}\mathcal{D}}\right)\right].\end{aligned} \tag{18}$$

Here, $\mathcal{L}(\mathcal{R}, \mathbf{0}) = \operatorname{tr}(\mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{G}_{\mathcal{R}\mathcal{R}}) - 2 \operatorname{tr}(\mathbf{C}_{\mathcal{R},:} \mathbf{G}_{:,\mathcal{R}}) + \text{const}$, where $\text{const} = (NT)^{-1} \sum_{q=1}^{N} \operatorname{tr}(\mathbf{G} \mathbf{X}_q^\top \mathbf{A}_\tau^\top \mathbf{A}_\tau \mathbf{X}_q)$ is independent of both $\mathcal{R}$ and $\mathbf{B}$. This damped objective is used for compensation in Eq. (12) and candidate scoring in Eq. (13); setting $\lambda_x = \lambda_w = 0$ recovers Eq. (11). This damping defines a numerically motivated regularized quadratic, rather than an equivalent reformulation of the original reconstruction loss. The term $\mathcal{L}(\mathcal{R}, \mathbf{0})$ must be retained when comparing different channel sets.

Differentiating Eq. (18) yields the optimality condition

$$\frac{1}{2} \nabla_{\mathbf{B}} \mathcal{L} = \mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{B} \mathbf{G}_{\mathcal{D}\mathcal{D}} - \mathbf{C}_{\mathcal{R},:} \mathbf{G}_{:,\mathcal{D}} + \mathbf{H}_{\mathcal{R}\mathcal{R}} \mathbf{G}_{\mathcal{R}\mathcal{D}} = \mathbf{0}. \tag{19}$$

Left-multiplying by $\mathbf{H}_{\mathcal{R}\mathcal{R}}^{-1}$ and right-multiplying by $\mathbf{G}_{\mathcal{D}\mathcal{D}}^{-1}$ gives Eq. (12). Positive damping makes both principal matrices positive definite, so this solution is the unique minimizer for a fixed retained set.

The damping coefficients are determined from the layer statistics and remain fixed throughout channel selection. Let $s_x = |\operatorname{tr}(\mathbf{H}_0)|/c$ and $s_w = \operatorname{tr}(\mathbf{W}^\top \mathbf{W})/c$, with each scale bounded below by the smallest positive normal FP64 value. We use

$$\lambda_x = c\epsilon s_x + \max\{\epsilon s_x, \epsilon\}, \qquad \lambda_w = c\epsilon s_w, \tag{20}$$

where $\epsilon = 2^{-23}$ is the machine epsilon of the FP32 calibration statistics. Linear solves are performed in FP64. The additional term in λ_x stabilizes the Cholesky factorization; neither coefficient requires hyperparameter tuning.

A.2 Efficient Reconstruction Updates

A candidate deletion changes both the retained inputs and the available compensation directions. For $j \in \mathcal{R}$, let $\mathcal{R}' = \mathcal{R} \setminus \{j\}$ and $\mathcal{D}' = \mathcal{D} \cup \{j\}$. We derive its compensated solution in two steps: first extend compensation to include $\mathbf{W}_{:,j}$ while retaining all current inputs, and then impose the removal of input channel j.

Let $\widetilde{\mathbf{B}} \in \mathbb{R}^{|\mathcal{R}|\times|\mathcal{D}'|}$ minimize the expanded quadratic obtained by replacing $\mathcal{D}$ with $\mathcal{D}'$ in Eq. (18), while keeping $\mathcal{R}$ and $\mathcal{L}(\mathcal{R}, \mathbf{0})$ fixed. Its solution follows from the same normal equation as Eq. (19). Comparing the columns corresponding to the previously dropped channels gives

$$\widetilde{\mathbf{B}}_{:,\mathcal{D}} = \mathbf{B}^\star_{\mathcal{R}} - \widetilde{\mathbf{B}}_{:,j}\, \mathbf{G}_{j,\mathcal{D}} \mathbf{G}^{-1}_{\mathcal{D}\mathcal{D}}. \tag{21}$$

Thus, adding the candidate weight direction requires one new coefficient column and a rank-one correction to the existing coefficients. The new column is obtained by eliminating the old columns through the Schur complement of $\mathbf{G}_{\mathcal{D}\mathcal{D}}$. When $\mathcal{D}$ is empty, the correction involving the old columns is absent.

Removing input channel j requires the corresponding row of the expanded coefficients to equal $-\mathbf{e}_j^\top$, where $\mathbf{e}_j \in \mathbb{R}^{|\mathcal{D}'|}$ is the unit vector associated with j. The minus sign cancels the original contribution of $\mathbf{W}_{:,j}$. Since $\widetilde{\mathbf{B}}$ minimizes the expanded quadratic, its first-order term vanishes. Minimizing the remaining quadratic change subject to this row constraint gives

$$\mathbf{B}^\star_{\mathcal{R}'} = \widetilde{\mathbf{B}}_{\mathcal{R}',:} - \frac{[\mathbf{H}^{-1}_{\mathcal{R}\mathcal{R}}]_{\mathcal{R}',j}}{[\mathbf{H}^{-1}_{\mathcal{R}\mathcal{R}}]_{jj}} \left(\widetilde{\mathbf{B}}_{j,:} + \mathbf{e}_j^\top\right). \tag{22}$$

This is a rank-one correction followed by removing row j. Substitution into the quadratic gives the exact candidate objective:

$$\begin{aligned}\mathcal{L}(\mathcal{R}', \mathbf{B}^\star_{\mathcal{R}'}) = \mathcal{L}(\mathcal{R}, \mathbf{0}) - \operatorname{tr}\Big(\mathbf{G}_{\mathcal{D}'\mathcal{D}'} \widetilde{\mathbf{B}}^\top \mathbf{H}_{\mathcal{R}\mathcal{R}} \widetilde{\mathbf{B}}\Big) \\ + \frac{\left(\widetilde{\mathbf{B}}_{j,:} + \mathbf{e}_j^\top\right) \mathbf{G}_{\mathcal{D}'\mathcal{D}'} \left(\widetilde{\mathbf{B}}_{j,:} + \mathbf{e}_j^\top\right)^\top}{[\mathbf{H}^{-1}_{\mathcal{R}\mathcal{R}}]_{jj}}.\end{aligned} \tag{23}$$

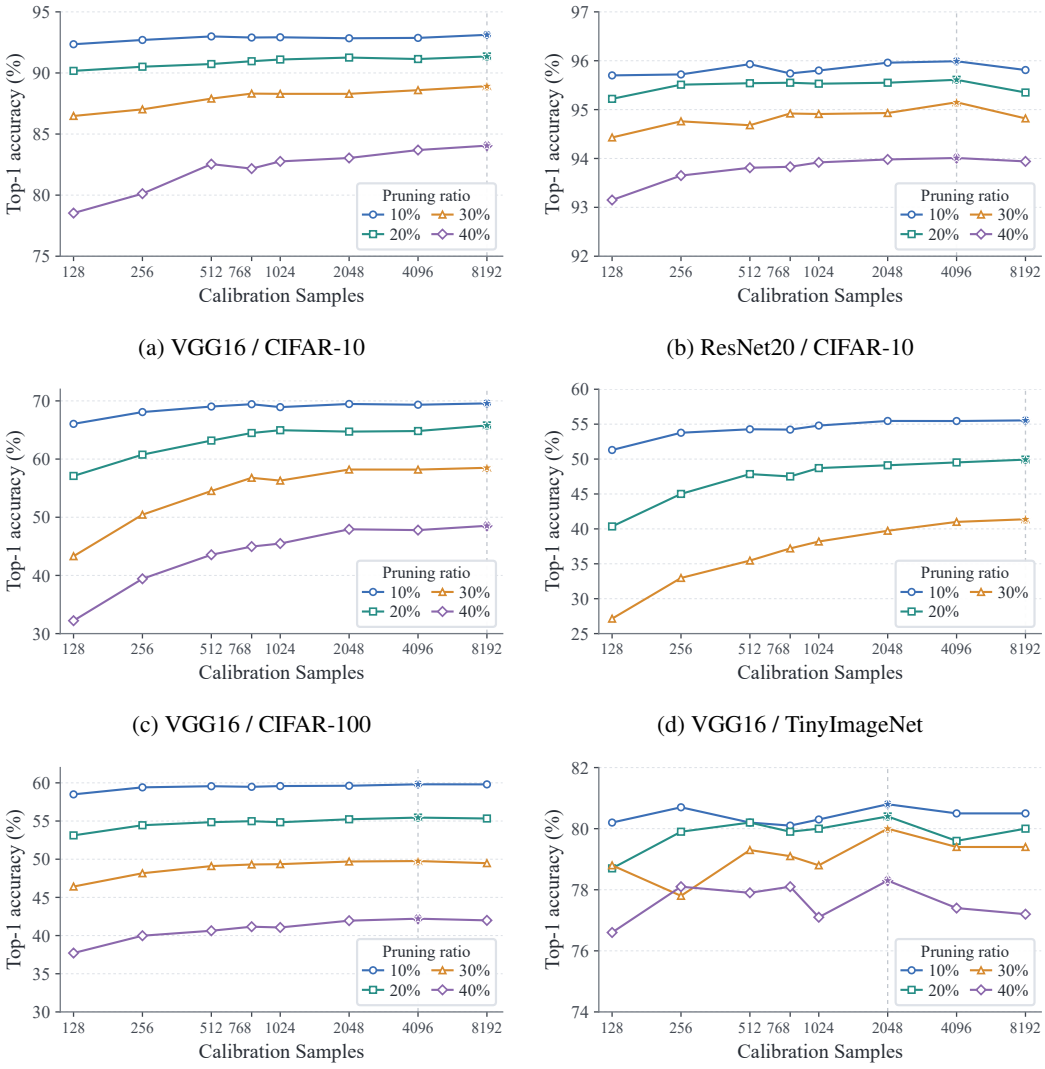
The first two terms give the minimum after expanding compensation, and the final term is the cost of enforcing input removal. Channel selection therefore accounts for both changes, rather than assessing removal with the compensation directions held fixed.

In implementation, we evaluate Eq. (23) from maintained residual correlations and quadratic forms, rather than explicitly constructing $\widetilde{\mathbf{B}}$ for every candidate. After accepting a deletion, the rank-one corrections in Eqs. (21)–(22), together with Schur-complement inverse updates, maintain the statistics required for subsequent scores. These updates involve matrix–vector products and outer products, giving $O(c^2)$ work per deletion on the rank-update path and $O((c-K)c^2)$ over the deletion sequence, excluding calibration, initialization, and numerical state rebuilds. Each candidate is evaluated at its fixed-structure optimum, while the overall channel-selection procedure remains greedy.

For TRIM, the same derivation applies using the moment representations $\mathbf{Z}_q$ and $\widehat{\mathbf{Z}}_q$ in the calibration statistics, with normalization $1/(2N)$. The compensation and selection updates are otherwise unchanged.

B Additional Experiments

B.1 Calibration-Set Size



(a) VGG16 / CIFAR-10

(b) ResNet20 / CIFAR-10

(c) VGG16 / CIFAR-100

(d) VGG16 / TinyImageNet

(e) SEW-ResNet18 / ImageNet-1K

(f) VGGSNN / DVS-CIFAR10

Figure 5: Accuracy versus calibration-set size for TRIM on additional visual model–dataset pairs. Stars mark the highest observed accuracy on each curve, and vertical dashed lines indicate the corresponding calibration-set sizes. Vertical-axis ranges vary across panels.

B.2 Compression and Energy Analysis

Energy estimation. The "Power" entries in Tables 2 and 5 report estimated arithmetic energy per sample, not measured device power. We count spike-triggered accumulations (ACs) and dense multiply–accumulates (MACs) for non-binary inputs, including SEW residual-sum inputs. Counts cover convolutional and linear layers and Spikingformer's attention products, summed over all executed time steps and averaged over the evaluation set. Using 32-bit floating-point costs in 45 nm CMOS (Horowitz, 2014), we estimate

$$E_{\text{est}} = 0.9\,\text{pJ}\,N_{\text{AC}} + 4.6\,\text{pJ}\,N_{\text{MAC}},$$

where N_{AC} and N_{MAC} denote these average operation counts; reported SOPs (M) equal $N_{\text{AC}}/10^6$.

Table 5: Additional efficiency metrics comparison of TRIM with full-precision uncompressed SNNs. Entries already reported in Table 2 are omitted. PR denotes the channel pruning ratio, and Power reports estimated inference energy.

	Architecture	PR (%)	Connection	Bit	Model size (MB)	SOPs (M)	Power (mJ)	Accuracy
CIFAR-10	VGG16	10	82.47%	32	48.56	138.31	0.154	93.12%
		20	66.68%	32	39.26	108.92	0.124	91.35%
		40	39.86%	32	23.47	58.85	0.072	84.05%
		50	29.04%	32	17.10	40.19	0.053	74.31%
	ResNet20	0	100%	32	68.17	759.19	0.700	96.36%
		10	90.00%	32	61.35	687.87	0.635	95.99%
		20	80.07%	32	54.59	610.54	0.566	95.61%
		30	69.95%	32	47.69	531.31	0.495	95.15%
		40	60.02%	32	40.92	449.34	0.421	94.01%
		50	50.02%	32	34.11	366.17	0.346	91.77%
CIFAR-100	VGG16	0	100%	32	59.06	186.16	0.201	71.52%
		10	82.52%	32	48.74	155.57	0.170	69.57%
		20	66.78%	32	39.45	120.57	0.135	65.78%
		30	52.45%	32	30.98	91.49	0.106	58.50%
		40	40.05%	32	23.66	64.59	0.078	48.53%
	ResNet20	10	90.02%	32	61.54	723.48	0.668	78.05%
		20	80.12%	32	54.77	640.09	0.593	76.37%
		30	70.03%	32	47.88	557.32	0.518	74.04%
		50	50.15%	32	34.29	386.08	0.364	64.11%
	Spikingformer	0	100%	32	37.30	472.80	0.450	79.09%
		10	91.74%	32	34.22	414.34	0.395	77.88%
		20	80.50%	32	30.03	335.45	0.321	73.19%
		30	69.42%	32	25.90	261.55	0.253	68.00%
TinyImageNet	VGG16	0	100%	32	59.27	834.27	0.883	59.08%
		10	82.58%	32	48.95	657.50	0.712	55.54%
		20	66.90%	32	39.65	499.91	0.556	49.92%
		30	52.61%	32	31.19	371.34	0.428	41.38%
ImageNet-1K	SEW-ResNet18	0	100%	32	46.72	465.28	13.018	63.27%
		10	90.59%	32	42.33	429.65	11.820	59.81%
		20	81.24%	32	37.96	387.83	10.597	55.45%
		30	71.73%	32	33.51	347.20	9.413	49.76%
		40	62.37%	32	29.14	301.11	8.186	42.21%
		50	52.97%	32	24.75	255.96	6.980	31.76%
DVS-CIFAR10	VGGSNN	10	83.40%	32	30.91	962.98	0.979	80.80%
		20	68.35%	32	25.34	772.81	0.795	80.40%
		30	54.52%	32	20.21	593.19	0.622	80.00%

B.3 Comparison with Training-Based Pruning

Table 6: Comparison with training-based SNN pruning methods.

Dataset	Method	Model	Training required	Accuracy (%)	Size (MB)
CIFAR-10	Chowdhury et al. (2021)	VGG9	✓	90.02	44.50
	Minimax (Chen et al., 2023)	6Conv2FC	✓	91.16	36.30†
	Shi et al. (2024)	6Conv2FC	✓	92.05	38.30†
	SCA (Li et al., 2024)	VGG16	✓	91.46	34.08†
	SCA (Li et al., 2024)	VGG16	✓	91.67	17.32†
	TRIM	VGG16	✗	91.35	39.26
	TRIM	ResNet20	✗	91.77	34.11
CIFAR-100	Chowdhury et al. (2021)	VGG11	✓	68.00	75.90
	SCA (Li et al., 2024)	VGG16	✓	67.11	47.16†
	TRIM	VGG16	✗	69.57	48.74
DVS-CIFAR10	Shi et al. (2024)	VGGSNN	✓	81.90	14.08†
	Shi et al. (2024)	VGGSNN	✓	78.30	7.24†
	TRIM	VGGSNN	✗	80.00	20.21
	TRIM	VGGSNN	✗	78.30	15.73

✓ indicates gradient-based training within the compression pipeline, including retraining, fine-tuning, or sparse training; ✗ indicates post-training pruning without such optimization. † denotes FP32 weight storage estimated from reported parameter counts or retention ratios, excluding sparse-index and mask overhead for the unstructured method of Shi et al. (2024). Repeated method–model entries correspond to different compression settings. Results are taken from the respective papers and our experiments; architectures, time steps, and training protocols are not matched across papers.

C EXPERIMENTAL SETTINGS

Table 7 summarizes the training configurations of the unpruned visual SNNs, including time steps, input resolution, batch size, optimizer, and learning-rate schedule. VGG16, ResNet20, and VGGSNN use the full-precision training recipe of QP-SNN (Wei et al., 2025), with SGD momentum 0.9 and temporal efficient training (TET) loss; the TET regularization coefficient and target mean are 10^{-3} and 1, respectively. SEW-ResNet18 uses cross-entropy loss and SGD with momentum 0.9. For Spikingformer, we report the default CIFAR training protocol of Zhou et al. (2023a). These settings concern the unpruned models; no post-pruning optimization is performed.

Table 7: Training configurations of the unpruned visual SNNs. In the CIFAR-10/100 column, the time steps 4 and 2 correspond to VGG16 and ResNet20, respectively. Batch size denotes the total training batch size.

Hyperparameter	CIFAR-10/100	TinyImageNet	ImageNet-1K	DVS-CIFAR10	CIFAR-100
Architecture	VGG16 / ResNet20	VGG16	SEW-ResNet18	VGGSNN	Spikingformer
Time steps	4 / 2	4	4	10	4
Resolution	32×32	64×64	224×224	48×48	32×32
Batch size	256	256	256	64	64
Training epochs	300	300	320	300	310
Optimizer	SGD	SGD	SGD	SGD	AdamW
Initial learning rate	0.1	0.1	0.1	0.1	5×10^{-4}
Weight decay	10^{-4}	10^{-4}	0	10^{-4}	6×10^{-2}
Learning-rate decay	Cosine	Cosine	Cosine	Cosine	Cosine

For VGG16 and ResNet20 on CIFAR, training uses random cropping, horizontal flipping, AutoAugment, and 16×16 Cutout. TinyImageNet uses random cropping and horizontal flipping, while ImageNet uses random resized cropping and horizontal flipping. For DVS-CIFAR10, events are integrated into ten frames with approximately equal event counts, resized to 48×48, and augmented with spatial cropping, horizontal flipping, and neuromorphic data augmentation; the dataset is split into 90% training and 10% test samples within each class. The official Spikingformer implementation specifies a 20-epoch learning-rate warm-up, a minimum learning rate of 10^{-5}, label smoothing of 0.1, Mixup with $\alpha = 0.5$, RandAugment, and random erasing with probability 0.25.