SmoothSpike: Spiking Transformer with Learnable Hadamard Transformation

Anonymous Authors[1]

Abstract

Spiking Neural Networks (SNNs) that leverage sparse binary spikes and temporal dynamics have emerged as energy-efficient alternatives to Artificial Neural Networks (ANNs). However, SNNs suffer from limited representational capacity due to the discrete nature of spikes. Existing solutions extending spike levels often overlook the constraints of the simulation time window, leading to a critical issue we identify as spike saturation-induced information homogenization. In this phenomenon, distinct high-amplitude inputs result in identical maximized spike counts, truncating the dynamic range and hindering the model's ability to capture fine-grained semantic differences. To address this, we propose SmoothSpike, a novel method designed to enhance representational capacity by suppressing spike saturation. We first introduce a randomized Hadamard transformation to smooth neuronal inputs, theoretically proving its efficacy in constraining extreme values and reducing both saturation probability and input variability among saturated neurons. To further improve adaptability, we evolve this into a learnable orthogonal transformation. Initialized with Hadamard matrices and maintained orthogonal via Newton-Schulz iteration, this module dynamically adapts to varying input distributions during training. Extensive experiments on language modeling tasks show that SmoothSpike effectively mitigates the information homogenization problem and improves task performance. This positions SmoothSpike as a robust solution to bridge the performance gap between SNNs and ANNs.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

1. Introduction

Brain-inspired spiking neural networks (SNNs) have gained increasing attention in recent years due to the spike-based communication and temporal dynamics (Gerstner & Kistler, 2002; Maass, 1997). Specifically, SNNs utilize sparse binary spikes to carry information and transmit information in a spike-driven manner, offering significant energy efficiency advantages. Moreover, the inherent temporal nature of SNNs allows for fine-grained representation of sequential data, making them well-suited for spatiotemporal modeling. As a result, a growing body of research is exploring the use of SNNs for complex language modeling tasks, such as SpikingBERT (Bal & Sengupta, 2024), SpikeLM (Xing et al., 2024b), and SpikeBERT (Lv et al., 2025). However, due to the discrete binary spikes, SNNs are less expressive in feature representation compared to Artificial Neural Networks (ANNs), resulting in a noticeable performance gap in complex language modeling tasks.

To narrow the performance gap between SNNs and ANNs, several studies have focused on improving the representational capacity of SNNs (Guo et al., 2022; Fang et al., 2023). For example, Guo et al. (2024); Xing et al. (2024b); Wang et al. (2025) extend traditional binary spikes to ternary spikes, introducing negative values to enhance the representation richness. In addition, Luo et al. (2024); Qiu et al. (2025) introduce multi-level spike states in the spike generation mechanism to provide more fine-grained discrete amplitudes. Despite the improved performance, these methods primarily enhance SNNs' representational capacity by expanding the set of possible spike values. Regretfully, these studies overlook the fact that within a limited time window, a single spiking neuron has a clear upper bound on the values it can represent, which directly limits its information discriminability and model representation capacity.

In SNNs, each spiking neuron has an upper limit T on the number of spikes it can emit due to the constrained time window. When the input value is large, the output spike count may reach this limit. As a result, different high-amplitude inputs may produce nearly identical output responses, making it difficult to distinguish input differences in the output space (Sava et al., 2023). We refer to this phenomenon as the spike saturation-induced information homogenization problem. This is equivalent to truncating the dynamic range

of the output, thereby compressing the representation of spiking neurons. This limitation hinders the ability of SNNs to capture fine-grained semantic differences. Therefore, addressing the spike saturation issue is crucial for improving SNN performance in complex language modeling tasks.

In this paper, we propose a method termed SmoothSpike, aimed at alleviating the spike saturation-induced information homogenization to improve performance. Specifically, we first analyze the spike saturation issue in spiking transformer architectures and experimentally reveal the resulting information homogenization phenomenon. To address this, we introduce a randomized Hadamard matrix to smooth the inputs of spiking neurons, reducing both the likelihood of neuron saturation and the dynamic range of inputs to saturated neurons. We then replace the randomized Hadamard matrix with a learnable orthogonal transformation matrix, which better adapts to varying input ranges, further enhancing the representational capacity of SNNs. We summarize the main contributions as follows:

- We systematically analyze the spike saturation issue, highlighting its widespread occurrence across layers and revealing the induced information homogenization phenomenon. This issue prevents the differences in input distribution from being reflected in the output space, thereby weakening the representational capacity of SNNs and impairing model performance.

- We propose using a randomized Hadamard transformation to smooth the inputs of spiking neurons and provide a theoretical proof that this transformation can constrain the maximum element of the input vector to a range of $\mathcal{O}(\sqrt{\frac{\log n}{n}})$. By applying this transformation, both the likelihood of spike saturation and the variance of inputs to saturated neurons are effectively reduced, thus preserving the model's representational capacity.

- We propose a learnable orthogonal transformation method, which is initialized with a randomized Hadamard matrix and approximates the orthogonal constraint using the Newton-Schulz iteration. This method allows the transformation matrix to adapt to varying input distributions during training, further enhancing the representational capacity of SNNs.

- Extensive experiments in complex language modeling show that our SmoothSpike significantly alleviates the spike saturation issue, thereby leading to substantial performance improvements across various tasks. Furthermore, detailed ablation validation confirms the effectiveness and generalizability of our method.

2. Related Works

2.1. SNNs for Language Modeling

Given the energy efficiency and temporal dynamics of SNNs, an increasing number of studies are exploring their application in language modeling tasks. For example, SpikingBERT (Bal & Sengupta, 2024) proposes a spiking language model that utilizes a spiking attention mechanism and knowledge distillation, achieving energy-efficient training and improved performance. SNN-BERT (Su et al., 2024) introduces a bidirectional parallel spiking neuron architecture and individual-based coding to build a parallel SNN, reducing memory overhead. Sorbet (Tang et al., 2024) proposes power-of-two softmax and bit-shifting powerNorm to replace traditional energy-intensive softmax and layer normalization, resulting in significant energy savings. SpikeLM (Xing et al., 2024b) introduces an elastic bi-spiking mechanism for language modeling, utilizing bidirectional spike encoding, spike frequency, and amplitude encoding to enhance performance. SpikeBERT (Lv et al., 2025) improves the Spikformer architecture for language tasks by introducing a two-stage knowledge distillation method, enabling energy-efficient text classification. Although these studies have improved performance, most have not analyzed and addressed the limited representation capability of spiking neurons, which is the primary reason for the lower performance of SNNs in language modeling tasks.

2.2. Spiking Neurons with High Expressive Capacity

Several studies have focused on improving the representational capacity of SNNs to enhance their task performance. For example, (Guo et al., 2024) proposes the ternary spike neuron, which extends binary spike activation to values of -1, 0, and 1, thereby increasing the information capacity of SNNs while maintaining their energy efficiency. (Wang et al., 2025) introduces a ternary spike-based neuromorphic signal processing system that uses a threshold-adaptive encoding method and a quantized ternary SNN, achieving energy-efficient signal processing and improved performance in tasks like keyword recognition and EEG identification. In addition to introducing negative values to enhance representational richness, some approaches use multi-level spike states to provide finer discrete amplitudes. For example, (Luo et al., 2024) presents the integer LIF neuron model, which reduces quantization errors during training with integer-valued activations and preserves spike-driven inference during testing, significantly improving object detection performance in SNNs. (Qiu et al., 2025) integrates an information-enhanced LIF neuron into a spiking transformer, achieving state-of-the-art results in various vision tasks. Despite the improvement, these studies overlook the fact that a single spiking neuron has a clear upper bound on the values it can represent, which directly limits its informa-

tion discriminability and model representation capacity

3. Preliminary

3.1. Hadamard Matrix

A Hadamard matrix of order n (denoted H_0) is an $n \times n$ square matrix with entries restricted to $\{+1, -1\}$ whose rows (and equivalently columns) are mutually orthogonal. This property is expressed by the relation

$$H_0^T H_0 = nI_n, \tag{1}$$

where I_n is the $n \times n$ identity matrix.

For many applications, particularly in signal processing, it is convenient to work with the normalized version of the Hadamard matrix, defined as

$$H = \frac{1}{\sqrt{n}} H_0. \tag{2}$$

This scaling ensures that H is an orthogonal matrix satisfying

$$H^T H = I_n \quad \text{and thus} \quad H^{-1} = H^T. \tag{3}$$

In the normalized form, every entry has absolute value $\frac{1}{\sqrt{n}}$.

Hadamard matrices are widely used in signal processing because the multiplication by H spreads a vector's energy uniformly across components, producing a flatter magnitude spectrum. In deep learning, QuIP# (Tseng et al., 2024) pioneered the use of randomized Hadamard transforms to smooth pre-activation distributions by spreading outlier energy, enabling more effective low-bit quantization of Large Language Models.

3.2. RMSNorm and Its Orthogonal Invariance

RMSNorm (Zhang & Sennrich, 2019) is a normalization method defined as:

$$\text{RMSNorm}(\mathbf{x}) = \frac{\mathbf{x}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}} \odot \boldsymbol{\gamma}, \tag{4}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^n$ is a learnable scaling parameter and $\odot$ denotes the Hadamard (element-wise) product.

When RMSNorm is used in a Pre-Norm architecture (Xiong et al., 2020), the learnable parameter $\boldsymbol{\gamma}$ can be absorbed into the subsequent spiking neurons' threshold terms. After this fusion, RMSNorm becomes orthogonally invariant. This property arises because, for any orthogonal matrix Q and any vector $\mathbf{x}$, orthogonal transformations preserve the Euclidean norm:

$$\|Q\mathbf{x}\|_2 = \|\mathbf{x}\|_2. \tag{5}$$

Consequently, the root-mean-square value computed in RMSNorm remains unchanged under orthogonal transformations of the input, making the normalization operation invariant to such rotations.

4. Methodology

4.1. Problem Analysis

Spiking Transformers transmit information through discrete spike sequences, offering advantages of low power consumption and event-driven computation. However, commonly used spiking neuron models, such as the Leaky Integrate-and-Fire (LIF) neuron, suffer from an inherent limitation known as the saturation problem.

The saturation problem refers to the fact that within a given time window T, a spiking neuron can emit at most T spikes. When the neuron receives large input values, even if there are significant differences among inputs, the output spike counts will all reach the upper limit T, resulting in severely degraded discrimination between different large input values. As illustrated in Fig 1a, this saturation phenomenon is prevalent across various layers of Spiking Transformer models, with a substantial proportion of neurons saturated. More critically, as shown in Fig 1b, the inputs to these saturated neurons exhibit considerable variability, indicating a significant loss of discriminative capacity.

Formally, let V_t denote the membrane potential of a neuron at time step t. Assuming consistent input at each time step, for inputs $x_1 > x_2 \gg 0$, we may observe that both yield the maximum spike count of T:

$$\sum_{t=1}^{T} s_t(x_1, V_{t-1}^{(1)}) = \sum_{t=1}^{T} s_t(x_2, V_{t-1}^{(2)}) = T, \tag{6}$$

where $s_t \in \{0, 1\}$ indicates whether a spike is emitted at time step t. This saturation phenomenon causes information loss, diminishes the network's capacity to represent input features, and consequently impacts model performance.

Fundamentally, the saturation problem stems from non-uniform input distributions—pre-activation values in certain dimensions are excessively large. Therefore, the key to addressing this problem lies in transforming the input to achieve a more uniform distribution, thereby avoiding saturation caused by extreme values while reducing the variance of inputs to saturated neurons.

4.2. Solution Based on Hadamard Transform

Method Overview To mitigate the saturation problem in spiking neurons, we propose applying a randomized Hadamard transform to the neuron inputs, i.e., multiplying the input by a Hadamard matrix before it enters the neurons. Let the input be $\mathbf{x} \in \mathbb{R}^n$ and the randomized Hadamard matrix be $H \in \mathbb{R}^{n \times n}$. The transformed input is:

$$\tilde{\mathbf{x}} = H\mathbf{x}. \tag{7}$$

This transformation effectively suppresses extreme values in the input, yielding a more uniform numerical distribution

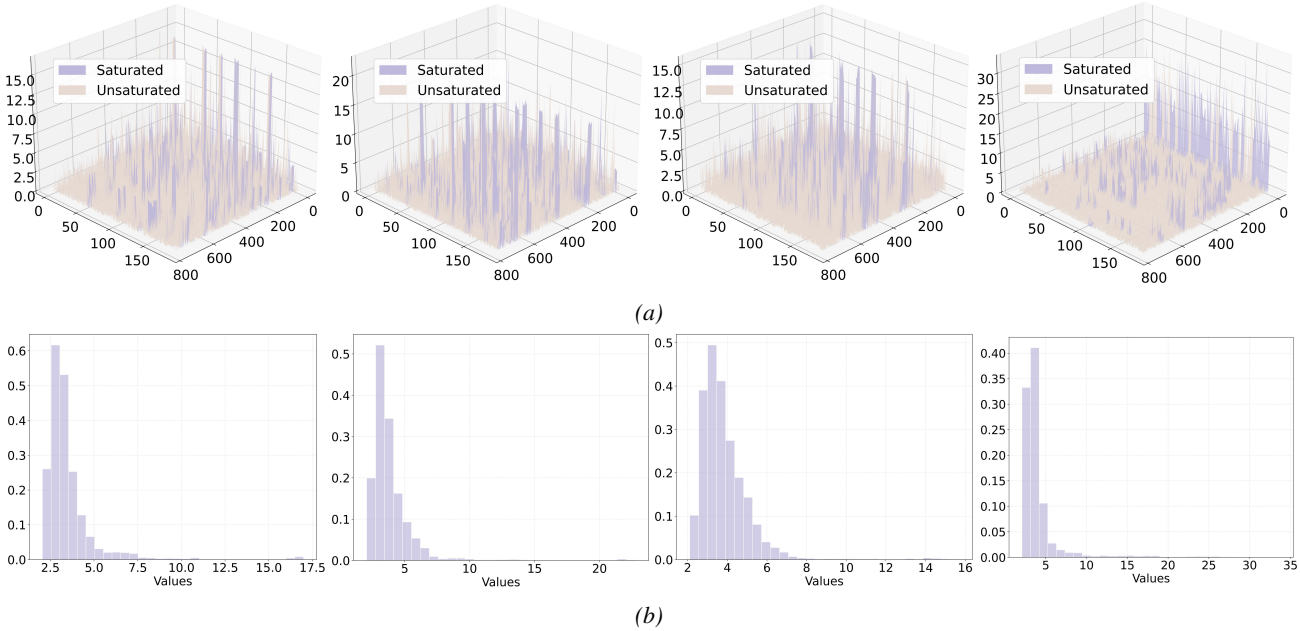

Figure 1. Saturation problem of neurons in Spiking Transformers. (a) Visualization of the inputs to neurons in two representative layers of the Spiking Transformer. (b) Density histograms of the inputs to saturated neurons in two representative layers.

and thereby reducing both the probability of neuron saturation and the dispersion of inputs among saturated neurons.

Intuitive Explanation As described in Subsection 3.1, the Hadamard matrix is a special orthogonal matrix where each element has absolute value $\frac{1}{\sqrt{n}}$ with sign $+1$ or -1. When the matrix dimension n is large (e.g., $n = 768$ as commonly seen in Spiking Transformers), $\frac{1}{\sqrt{n}}$ becomes quite small, approximately 0.04. When the input vector is multiplied by a Hadamard matrix, each component of the transformed vector becomes a weighted sum of all components of the original vector, with weights $\pm\frac{1}{\sqrt{n}}$. This means that large values originally concentrated in a few dimensions are dispersed across all dimensions. Moreover, due to the alternating positive and negative signs, these contributions partially cancel during summation, effectively smoothing out the extreme values in the original input.

Theoretical Guarantees

Theorem 4.1. *(Maximum Element Bound for Randomized Hadamard Transform) Let $U \in \mathbb{R}^{n\times n}$ be an orthogonal Hadamard matrix, and let S be a random diagonal matrix whose diagonal entries are independent and uniformly distributed over $\{+1, -1\}$. For any fixed vector $\mathbf{x} \in \mathbb{R}^n$, let $\mathbf{e}_i \in \mathbb{R}^n$ denote the i-th standard basis vector. Then for any $\epsilon > 0$:*

$$P\left(\max_i \left|\mathbf{e}_i^T US\mathbf{x}\right| \geq \|\mathbf{x}\|_2\sqrt{\frac{2}{n}\log\frac{2n}{\epsilon}}\right) \leq \epsilon. \quad (8)$$

Theorem 4.1[1] provides rigorous theoretical justification for the Hadamard transform's effectiveness in mitigating the saturation problem. The theorem demonstrates that after the randomized Hadamard transform US, the maximum element of the output vector is bounded by $O\left(\sqrt{\frac{\log n}{n}}\right)$ with high probability.

To better understand how this theorem addresses the saturation problem, consider an extreme case. Suppose a component of the input vector $\mathbf{x}$ is very large, approaching the total energy of the vector (i.e., close to $\|\mathbf{x}\|_2$). Without transformation, this large value would directly cause the corresponding neuron to saturate. However, according to Theorem 4.1, the maximum element of the transformed vector does not exceed $\|\mathbf{x}\|_2\sqrt{\frac{2\log(2n/\epsilon)}{n}}$ with probability $1-\epsilon$. For typical dimension $n = 768$ and confidence level $\epsilon = 0.01$, this bound is approximately $0.13\|\mathbf{x}\|_2$, far smaller than the extreme values that appear before transformation.

Inverse Transform To enable the output of spiking neurons to be properly processed by subsequent network layers, we apply the inverse Hadamard transform at the neuron output, recovering the representation from the Hadamard domain to the original domain. Since the Hadamard matrix is orthogonal, its inverse equals its transpose, i.e., $H^{-1} = H^T$. Let the output spike sequence of the neuron be $\mathbf{s}$, then the final output is:

$$\mathbf{y} = H^T\mathbf{s}. \quad (9)$$

[1]The proof of Theorem 4.1 is provided in Appendix A.1.

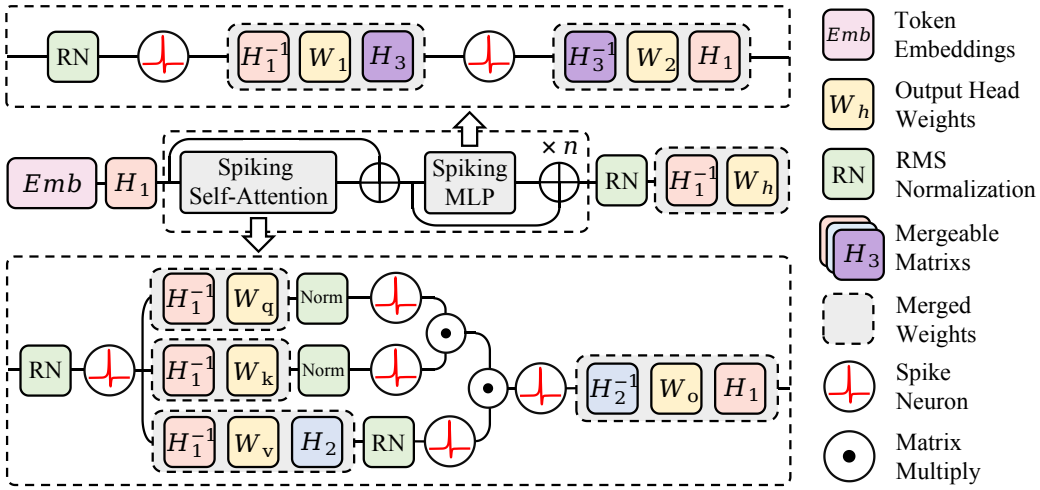

Figure 2. Overall architecture of our method, illustrated using the Spikingformer framework as an example. To adapt the Spikingformer architecture to our method, we replaced Batch Normalization with RMS Normalization and incorporated the PreNorm structure.

From a macroscopic perspective, the Hadamard transform and its inverse constitute an encoding-decoding pair: the encoding stage transforms the input into a space more suitable for spiking neuron processing, while the decoding stage restores the output to the original space, enabling seamless integration with subsequent network layers.

4.3. Overall Architecture

The overall architecture is designed based on the methods described above, as illustrated in Fig. 2. A core design objective of this architecture is that, during the inference phase, all Hadamard transform matrices (except the first one) can be fused into the model weights. This approach introduces negligible additional computational overhead and preserves the event-driven computation paradigm of the model during inference.

The matrix H_3 in the figure can be directly fused into the model weights. However, fusing H_1 and H_2 relies on the orthogonal invariance property of RMSNorm without learnable weight (denoted as RMSNorm$'$) presented in Subsection 3.2. Specifically, we can imagine absorbing the scaling weight of RMSNorm into the threshold of the subsequent spiking neurons during training, and then the participation of H_1 (as with H_2) in the forward computation can be expressed as:

$$\mathbf{z} = H_1 \cdot \text{RMSNorm}'(W_l \mathbf{x}). \tag{10}$$

Due to the orthogonal invariance property of RMSNorm$'$, the Hadamard transform and normalization operations commute, yielding the equivalent form

$$\mathbf{z} = \text{RMSNorm}'(H_1 W_l \mathbf{x}) = \text{RMSNorm}'(\tilde{W}_l \mathbf{x}), \tag{11}$$

where $\tilde{W}_l = H_1 W_l$ represents the fused weight matrix. This equivalence demonstrates that the orthogonal matrices H_1 and H_2 can be pre-fused into the weights of the corresponding linear layers during inference. Similarly, the inverse transform H^T applied after the spiking neurons can be absorbed into the weights of the subsequent linear layer.

It should be noted that we do not apply the Hadamard transform to the inputs of the LIF neurons responsible for generating the query and key vectors, as experimental results show that these neurons exhibit only mild saturation. Furthermore, owing to the relatively high dimensionality of H_3, we constrain it to a block-diagonal structure consisting of four equally sized blocks, thereby reducing the computational cost during training.

4.4. Learnable Orthogonal Transform Matrix

Although a fixed randomized Hadamard matrix can effectively mitigate the saturation problem, for different input distribution characteristics, a fixed matrix is not optimal. Intuitively, if we use the randomized Hadamard matrix as initialization and subsequently learn an optimal transform matrix that adapts to the actual distribution characteristics of the data during training, further performance improvement can be expected. The transform matrix must remain orthogonal throughout training, as orthogonality is required for commuting with RMSNorm$'$ and enabling seamless fusion into the weights during inference.

To achieve learnable orthogonal matrices, we employ the matrix sign function (msign) for orthogonal projection. Specifically, we initialize the learnable matrix W with a randomized Hadamard matrix. During forward propagation, we apply the msign function to W, projecting it onto an orthogonal matrix before it participates in computation.

Definition 4.2. (Matrix Sign Function) For a matrix $A \in \mathbb{R}^{n \times n}$, let its singular value decomposition be $A = U\Sigma V^T$.

The msign is defined as:

$$\text{msign}(A) = UV^T. \tag{12}$$

That is, the left and right singular vectors of A are retained while all singular values are replaced with 1.

In practice, we employ a fifth-order variant of the Newton-Schulz iteration to approximate the matrix sign function (Bernstein & Newhouse, 2024; Liu et al., 2025). Initialize $Y_0 = W/\|W\|_F$, where $\|W\|_F$ denotes the Frobenius norm, ensuring the singular values of Y_0 lie in $(0, \sqrt{3})$. The iteration proceeds as:

$$Y_{k+1} = aY_k + b\,Y_k(Y_k^TY_k) + c\,Y_k(Y_k^TY_k)^2, \tag{13}$$

After K iterations, Y_k serves as an orthogonal approximation of W.

This iteration offers two key advantages. First, it is significantly more computationally efficient than direct singular value decomposition, requiring lower execution time and fewer resources. Second, its computational graph is fully differentiable, allowing seamless integration with automatic differentiation in modern deep learning frameworks.

A detailed explanation of how the Newton-Schulz iteration approximates the matrix sign function, as well as the mechanism by which gradients are backpropagated to W, is provided in Appendix B.

Theorem 4.3. *(Optimal Orthogonal Approximation Property of Matrix Sign Function) For any full-rank matrix $A \in \mathbb{R}^{n\times n}$, its matrix sign function $Q = \text{msign}(A)$ is the optimal orthogonal approximation of A under the Frobenius norm, i.e.:*

$$Q = \text{msign}(A) = \arg\min_{Q'\in\mathcal{O}(n)} \|A - Q'\|_F, \tag{14}$$

where $\mathcal{O}(n)$ denotes the n-dimensional orthogonal matrix group.

Theorem 4.3[2] guarantees that the orthogonal matrix obtained through the msign function is the optimal orthogonal approximation of the original learnable matrix. This means that during training, gradient updates adjust the original matrix W, while the msign function ensures that what actually participates in computation is always the orthogonal matrix closest to W. This mechanism maintains optimization flexibility while strictly satisfying the orthogonality requirement.

In summary, we initialize H_1, H_2, and H_3 in Fig. 2 with randomized Hadamard matrices and set them as learnable parameters. During forward propagation, we apply the msign function approximated via Newton-Schulz iteration to ensure that H_1, H_2, and H_3 remain orthogonal. During backpropagation, these matrices are updated through the automatic differentiation mechanism of the deep learning framework. We denote this method as SmoothSpike.

[2]The proof of Theorem 4.3 is provided in Appendix A.2.

5. Experiments

5.1. Performance on the GLUE Benchmark

To evaluate the effectiveness of our proposed SmoothSpike method, we conduct extensive experiments on the GLUE (General Language Understanding Evaluation) benchmark (Wang et al., 2018), which comprises eight diverse natural language understanding tasks: MNLI (Multi-Genre Natural Language Inference), QQP (Quora Question Pairs), QNLI (Question Natural Language Inference), SST-2 (Stanford Sentiment Treebank), CoLA (Corpus of Linguistic Acceptability), STS-B (Semantic Textual Similarity Benchmark), MRPC (Microsoft Research Paraphrase Corpus), and RTE (Recognizing Textual Entailment). Detailed experimental settings are provided in Appendix C.

We integrate SmoothSpike into widely used spiking Transformer baselines, and compare their performances against vanilla baselines, as well as other representative SNNs, ANNs, and quantized neural networks (QNNs). The comparison includes $\text{BERT}_{\text{base}}$ (Devlin et al., 2019) as a high-performance ANN reference, ELMo (Peters et al., 2018) as an embedding-based baseline, and various quantized and spiking models such as BiBERT (Qin et al., 2022), BiT (Liu et al., 2022), BiPFT (Xing et al., 2024a), SpikeBERT, PSN-BERT, LIF-BERT (Xing et al., 2024b), Spikingformer (Zhou et al., 2023) and SpikeLM.

As shown in Table 1, the vanilla Spikingformer achieves an average score of 66.8 across the GLUE tasks. Incorporating SmoothSpike yields substantial improvements, resulting in an average score of 75.0, which represents a gain of 8.2 points. Moreover, meaningful performance improvements were achieved across all tasks in GLUE. Notable enhancements are observed in tasks requiring fine-grained semantic discrimination, such as CoLA (+16.8 points), STS-B (+24.8 points), and QNLI (+7.8 points). These gains underscore SmoothSpike's ability to alleviate spike saturation and enhance representational capacity. Similar improvements are observed for SpikeLM, another baseline model, though with reduced magnitude compared to Spikingformer. This difference arises because SpikeLM already utilizes advanced mechanisms such as the elastic bi-Spiking mechanism to substantially enhance its representational capacity, thereby limiting the additional gains that SmoothSpike can provide.

5.2. Ablation Study

To investigate the individual contributions of the components in SmoothSpike, we conduct ablation experiments on

Table 1. Comparison of performance on the GLUE development set for the baseline Spiking Transformer with and without SmoothSpike, alongside various other SNNs, ANNs and QNNs. T refers to time steps of the model. The unit of energy is mJ.

Model	T	Energy	MNLI$_{\text{-m/mm}}$	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
BERT$_{\texttt{base}}$	–	51.41	83.8/83.4	90.5	90.7	92.3	60.0	89.4	89.8	69.3	83.2
ELMo	–	–	68.6/–	86.2	71.1	91.5	44.1	70.4	76.6	53.4	70.2
BiBERT	–	–	66.1/67.5	84.8	72.6	88.7	25.4	33.6	72.5	57.4	63.2
BiT	–	–	77.1/77.5	82.9	85.7	87.7	25.1	71.1	79.7	58.8	71.0
BiPFT	–	–	69.5/70.6	83.7	81.7	86.2	22.9	80.2	76.2	66.1	70.8
SpikeBERT	4	14.30	71.4/71.0	68.2	66.4	85.4	16.9	18.7	82.0	57.5	59.7
PSN-BERT	4	–	35.4/35.2	0.0	50.5	50.9	0.0	6.8	81.2	52.7	34.7
LIF-BERT	4	7.98	56.8/55.2	70.0	60.6	80.6	14.6	20.0	82.3	53.8	54.9
Spikingformer	4	6.76	71.9/72.5	84.7	76.0	87.2	24.4	54.5	79.7	55.6	66.8
+SmoothSpike	4	9.45	**75.1/76.1**	**87.6**	**83.8**	**88.6**	**41.2**	**79.3**	**84.8**	**58.5**	**75.0**
SpikeLM	1	3.98	76.0/76.9	84.0	84.9	86.5	37.9	**84.3**	85.6	**65.3**	75.7
+SmoothSpike	1	7.03	**76.8/77.7**	**84.3**	**86.8**	**89.5**	**52.7**	83.5	**88.1**	58.1	**77.5**

Table 2. Ablation results for individual components of SmoothSpike on the GLUE development set. $Learnable$ indicates whether the Hadamard matrix is set as learnable.

Method				MNLI$_{\text{-m/mm}}$	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	Avg.
H_1	H_2	H_3	$Learnable$									
				67.13/67.37	76.82	70.73	83.72	19.10	51.84	82.27	57.04	64.00
✓				67.83/68.45	78.19	71.13	83.03	19.24	50.55	81.48	53.79	63.74
✓			✓	70.21/69.69	79.45	75.51	84.75	16.07	61.29	81.79	54.15	65.88
✓	✓		✓	71.16/71.30	80.24	77.06	86.01	22.30	64.18	82.04	54.51	67.64
✓	✓	✓	✓	71.20/71.40	80.92	78.58	85.55	19.89	65.26	82.15	55.96	**67.88**

the GLUE development set using a 4-layer Spikingformer model with a hidden dimension of 768. The results are presented in Table 2, where we systematically incorporate the Hadamard transformations H_1, H_2, and H_3, and evaluate the impact of making these matrices learnable.

The baseline without any transformations achieves an average score of 64.0. Incorporating the learnable H_1 improves the average to 65.88 (+1.88 points), demonstrating its effectiveness in smoothing inputs to mitigate saturation. The successive addition of learnable H_2 and H_3 further boosts performance to 67.64 (+3.64 points) and 67.88 (+3.88 points), respectively, indicating the complementary benefits of applying the transformation at multiple positions.

To validate the efficacy of our proposed learnable orthogonal matrix strategy, we compare two configurations for H_1: (1) a fixed randomized Hadamard matrix, and (2) a learnable matrix initialized with a randomized Hadamard matrix. The learnable H_1 outperforms its fixed counterpart by 2.14 points in average score, underscoring the advantage of adapting the transformation to input data distributions.

Notably, the fixed H_1 configuration achieves slightly lower average performance than the baseline. Specifically, while the fixed matrix surpasses the baseline on complex tasks such as MNLI, QQP, and QNLI—indicating some capacity for saturation suppression—it underperforms significantly on simpler tasks including MRPC and RTE. This disparity suggests that although a fixed Hadamard matrix can alleviate saturation to some extent, its lack of adaptability to varying inputs limits generalization and leads to overfitting on specific task types.

In summary, these ablation results confirm that each component contributes meaningfully to the overall performance, with learnable parameters providing additional gains through enhanced adaptability to data distributions.

5.3. Effect of SmoothSpike

Figure 3 illustrates the distributional shifts of pre-activation values across four representative layers in Spikingformer with SmoothSpike, contrasting states before and after the learnable transformation. The upper row depicts pre-transformation distributions, while the lower row presents the corresponding post-transformation distributions.

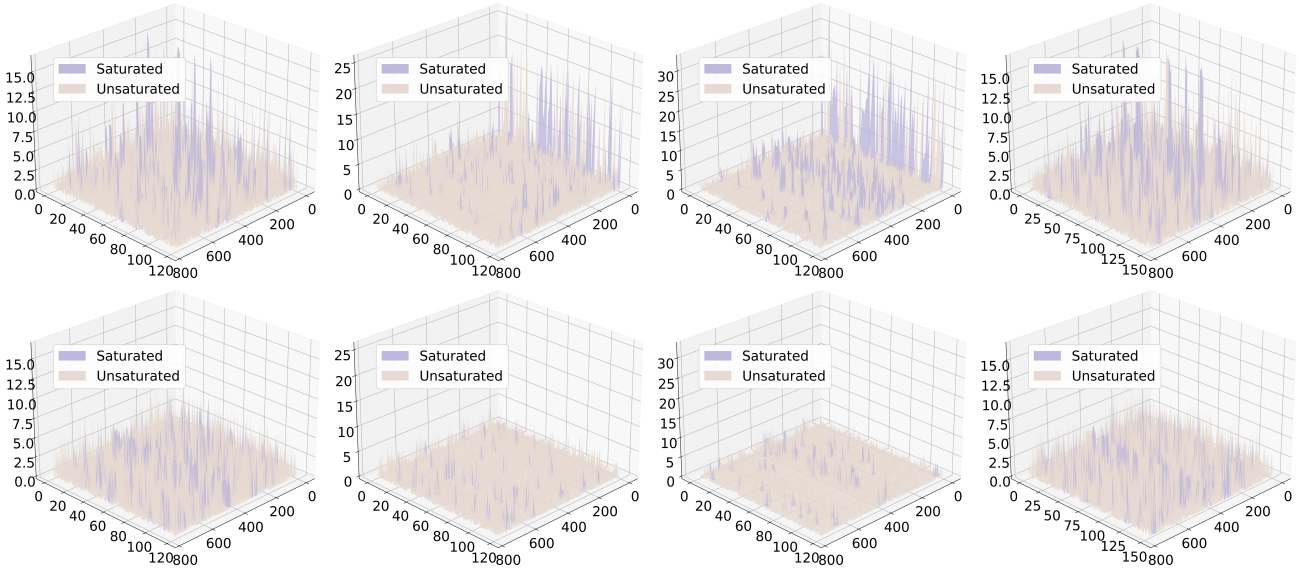

Figure 3. Comparison of pre-activation value distributions before and after applying the learnable Hadamard transform. The first row illustrates the distributions of pre-activations from four specific layers prior to the learnable Hadamard transform, while the second row depicts the distributions for the corresponding layers after the transform.

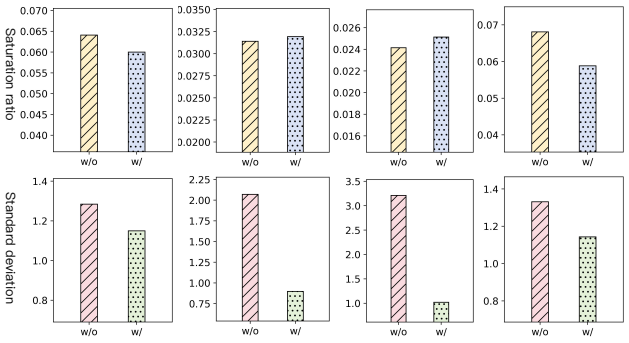

Figure 4. Comparison of the proportion of saturated neurons (first row) and the standard deviation of inputs to saturated neurons (second row) when using (w/) versus not using (w/o) the learnable Hadamard transform.

Notably, prior to transformation, pre-activation values exhibit numerous outliers substantially deviating from the normal range, causing neuron saturation. These saturated neurons cannot effectively discriminate among such highly divergent outlier inputs, degrading the model's representational capacity. By contrast, the transformation markedly smooths these distributions, yielding more uniform distributions (as shown in the second row of Figure 3).

More specifically, as illustrated in Figure 4, when the proportion of saturated neurons is relatively high, the transformation effectively reduces both the saturation ratio and the variance of inputs to saturated neurons, thereby enhancing representational capacity. Conversely, when saturation ratios are low, the transformation substantially reduces the variance of inputs to saturated neurons while maintaining basically constant saturation ratios. Although the proportion of neurons outputting the maximum value T remains comparable, the inputs triggering these T outputs become more concentrated, enabling finer-grained discrimination among previously divergent inputs and thus enhancing representational capacity.

In conclusion, the learnable Hadamard transformation effectively mitigates spike saturation-induced information homogenization by smoothing pre-activation values.

6. Conclusion

This work identifies spike saturation-induced information homogenization as a critical limitation in Spiking Neural Networks, where bounded temporal windows cause distinct high-amplitude inputs to converge to identical spike counts, compressing representational capacity. To address this issue, we propose SmoothSpike, a novel framework that employs randomized Hadamard transformations to suppress extreme input values, complemented by a learnable orthogonal module adapted via Newton-Schulz iterations to accommodate varying input distributions. We theoretically prove the efficacy of the smoothing mechanism in constraining maximum input magnitudes. Extensive experiments on language modeling tasks demonstrate that SmoothSpike consistently mitigates information homogenization while maintaining the energy efficiency inherent to binary spike-based computation. These results establish that addressing input saturation through orthogonal smoothing offers a principled pathway toward bridging the performance gap between energy-efficient SNNs and conventional ANNs.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Bal, M. and Sengupta, A. Spikingbert: Distilling bert to train spiking language models using implicit differentiation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 10998–11006, 2024.

Bentivogli, L., Clark, P., Dagan, I., and Giampiccolo, D. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1, 2009.

Bernstein, J. and Newhouse, L. Old optimizer, new norm: An anthology, 2024. URL https://arxiv.org/abs/2409.20325.

Björck, Å. and Bowie, C. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364, 1971.

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., and Specia, L. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Dagan, I., Glickman, O., and Magnini, B. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pp. 177–190. Springer, 2005.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pp. 4171–4186, 2019.

Dolan, W. B. and Brockett, C. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the third international workshop on paraphrasing (IWP2005)*, 2005.

Fang, W., Yu, Z., Zhou, Z., Chen, D., Chen, Y., Ma, Z., Masquelier, T., and Tian, Y. Parallel spiking neurons with high efficiency and ability to learn long-term dependencies. *Advances in Neural Information Processing Systems*, 36:53674–53687, 2023.

Gerstner, W. and Kistler, W. M. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

Giampiccolo, D., Magnini, B., Dagan, I., and Dolan, W. B. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pp. 1–9, 2007.

Guo, Y., Zhang, L., Chen, Y., Tong, X., Liu, X., Wang, Y., Huang, X., and Ma, Z. Real spike: Learning real-valued spikes for spiking neural networks. In *European Conference on Computer Vision*, pp. 52–68. Springer, 2022.

Guo, Y., Chen, Y., Liu, X., Peng, W., Zhang, Y., Huang, X., and Ma, Z. Ternary spike: Learning ternary spikes for spiking neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 12244–12252, 2024.

Haim, R. B., Dagan, I., Dolan, B., Ferro, L., Giampiccolo, D., Magnini, B., and Szpektor, I. The second pascal recognising textual entailment challenge. In *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*, volume 7, pp. 785–794, 2006.

Higham, N. J. *Functions of matrices: theory and computation*. SIAM, 2008.

Kovarik, Z. Some iterative methods for improving orthonormality. *SIAM Journal on Numerical Analysis*, 7(3):386–389, 1970.

Liu, J., Su, J., Yao, X., Jiang, Z., Lai, G., Du, Y., Qin, Y., Xu, W., Lu, E., Yan, J., et al. Muon is scalable for llm training. *arXiv preprint arXiv:2502.16982*, 2025.

Liu, Z., Oguz, B., Pappu, A., Xiao, L., Yih, S., Li, M., Krishnamoorthi, R., and Mehdad, Y. Bit: Robustly binarized multi-distilled transformer. *Advances in neural information processing systems*, 35:14303–14316, 2022.

Luo, X., Yao, M., Chou, Y., Xu, B., and Li, G. Integer-valued training and spike-driven inference spiking neural network for high-performance and energy-efficient object detection. In *European Conference on Computer Vision*, pp. 253–272. Springer, 2024.

Lv, C., Li, T., Qiao, W., Wang, X., Wu, M., Liu, W., Dou, S., Zheng, X., and Huang, X. Spikebert: A language spikformer learned from bert with knowledge distillation. *Neural Networks*, pp. 108482, 2025.

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

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep contextualized word representations, 2018. URL https://arxiv.org/abs/1802.05365.

Qin, H., Ding, Y., Zhang, M., Yan, Q., Liu, A., Dang, Q., Liu, Z., and Liu, X. Bibert: Accurate fully binarized bert. *arXiv preprint arXiv:2203.06390*, 2022.

Qiu, X., Zhang, M., Zhang, J., Wei, W., Cao, H., Guo, J., Zhu, R.-J., Shan, Y., Yang, Y., and Li, H. Quantized spike-driven transformer. In *The Thirteenth International Conference on Learning Representations*, 2025.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

Sava, R., Donati, E., and Indiveri, G. Feed-forward and recurrent inhibition for compressing and classifying high dynamic range biosignals in spiking neural network architectures. In *2023 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 1–5. IEEE, 2023.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Su, Q., Mei, S., Xing, X., Yao, M., Zhang, J., Xu, B., and Li, G. Snn-bert: Training-efficient spiking neural networks for energy-efficient bert. *Neural Networks*, 180:106630, 2024.

Tang, K., Yan, Z., and Wong, W.-F. Sorbet: A neuromorphic hardware-compatible transformer-based spiking language model. *arXiv preprint arXiv:2409.15298*, 2024.

Tseng, A., Chee, J., Sun, Q., Kuleshov, V., and De Sa, C. Quip#: Even better llm quantization with hadamard incoherence and lattice codebooks. *arXiv preprint arXiv:2402.04396*, 2024.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353. Association for Computational Linguistics, 2018.

Wang, S., Zhang, D., Belatreche, A., Xiao, Y., Qing, H., Wei, W., Zhang, M., and Yang, Y. Ternary spike-based neuromorphic signal processing system. *Neural Networks*, 187:107333, 2025.

Warstadt, A., Singh, A., and Bowman, S. R. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Williams, A., Nangia, N., and Bowman, S. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long papers)*, pp. 1112–1122, 2018.

Xing, X., Du, L., Wang, X., Zeng, X., Wang, Y., Zhang, Z., and Zhang, J. Bipft: Binary pre-trained foundation transformer with low-rank estimation of binarization residual polynomials. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16094–16102, 2024a.

Xing, X., Zhang, Z., Ni, Z., Xiao, S., Ju, Y., Fan, S., Wang, Y., Zhang, J., and Li, G. Spikelm: Towards general spike-driven language modeling via elastic bi-spiking mechanisms. In *International Conference on Machine Learning*, pp. 54698–54714. PMLR, 2024b.

Xiong, R., Yang, Y., He, D., Zheng, K., Zheng, S., Xing, C., Zhang, H., Lan, Y., Wang, L., and Liu, T. On layer normalization in the transformer architecture. In *International conference on machine learning*, pp. 10524–10533. PMLR, 2020.

Zhang, B. and Sennrich, R. Root mean square layer normalization. *Advances in neural information processing systems*, 32, 2019.

Zhou, C., Yu, L., Zhou, Z., Ma, Z., Zhang, H., Zhou, H., and Tian, Y. Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. *arXiv preprint arXiv:2304.11954*, 2023.

A. Proofs of Theorems

A.1. Proof of Theorem 4.1

Lemma 1 *(Hoeffding's Inequality) Let $Z_1, Z_2, \ldots, Z_n$ be independent random variables with $Z_i \in [a_i, b_i]$ almost surely. Define $S = \sum_{i=1}^{n} Z_i$. Then for any $t > 0$:*

$$P(|S - \mathbb{E}[S]| \geq t) \leq 2\exp\left(-\frac{2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right) \tag{15}$$

We establish the concentration bound by analyzing the distribution of each coordinate under the randomized Hadamard transform, then applying a union bound.

For a fixed index $i \in \{1, 2, \ldots, n\}$, consider:

$$Y_i = \mathbf{e}_i^T USx = \sum_{j=1}^{n} U_{ij} x_j S_{jj} \tag{16}$$

where U_{ij} and x_j are deterministic constants, and $S_{jj} \in \{+1, -1\}$ are independent Rademacher random variables.

Define $c_j = U_{ij}x_j$ for each j. Then:

$$Y_i = \sum_{j=1}^{n} c_j S_{jj} \tag{17}$$

This is a weighted sum of independent Rademacher variables. Since $\mathbb{E}[S_{jj}] = 0$, we have:

$$\mathbb{E}[Y_i] = \sum_{j=1}^{n} c_j \mathbb{E}[S_{jj}] = 0 \tag{18}$$

Each term $c_j S_{jj}$ takes values in $[-|c_j|, +|c_j|]$, so $b_j - a_j = 2|c_j|$. By Lemma 1:

$$P(|Y_i| \geq a) \leq 2\exp\left(-\frac{2a^2}{\sum_{j=1}^{n}(2|c_j|)^2}\right) = 2\exp\left(-\frac{a^2}{2\sum_{j=1}^{n} c_j^2}\right) \tag{19}$$

Since U is an orthogonal Hadamard matrix, each entry satisfies $U_{ij} = \pm\frac{1}{\sqrt{n}}$. Therefore:

$$\sum_{j=1}^{n} c_j^2 = \sum_{j=1}^{n}(U_{ij}x_j)^2 = \sum_{j=1}^{n} \frac{x_j^2}{n} = \frac{1}{n}\sum_{j=1}^{n} x_j^2 = \frac{\|\mathbf{x}\|_2^2}{n} \tag{20}$$

Substituting this into the Hoeffding bound:

$$P(|Y_i| \geq a) \leq 2\exp\left(-\frac{a^2}{2 \cdot \frac{\|\mathbf{x}\|_2^2}{n}}\right) = 2\exp\left(-\frac{na^2}{2\|\mathbf{x}\|_2^2}\right) \tag{21}$$

Applying the union bound over all n coordinates:

$$P\left(\max_{i\in\{1,\ldots,n\}} |Y_i| \geq a\right) \leq \sum_{i=1}^{n} P(|Y_i| \geq a) \leq 2n\exp\left(-\frac{na^2}{2\|\mathbf{x}\|_2^2}\right) \tag{22}$$

To ensure this probability is at most ϵ, we require:

$$2n\exp\left(-\frac{na^2}{2\|\mathbf{x}\|_2^2}\right) \leq \epsilon \tag{23}$$

Taking logarithms and rearranging:

$$\frac{na^2}{2\|\mathbf{x}\|_2^2} \geq \log\left(\frac{2n}{\epsilon}\right) \tag{24}$$

Solving for a:

$$a^2 \geq \frac{2\|\mathbf{x}\|_2^2}{n} \log\left(\frac{2n}{\epsilon}\right) \tag{25}$$

Therefore:

$$a = \|\mathbf{x}\|_2 \sqrt{\frac{2}{n} \log\left(\frac{2n}{\epsilon}\right)} \tag{26}$$

This completes the proof.

A.2. Proof of Theorem 4.3

Let $A \in \mathbb{R}^{n\times n}$ be a full-rank matrix with singular value decomposition:

$$A = U\Sigma V^T \tag{27}$$

where $U, V \in \mathcal{O}(n)$ are orthogonal matrices and $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ is a diagonal matrix with $\sigma_i > 0$ for all i.

For any orthogonal matrix $Q' \in \mathcal{O}(n)$, we minimize:

$$\|A - Q'\|_F^2 = \text{tr}[(A - Q')^T(A - Q')] \tag{28}$$

Expanding the trace expression yields:

$$\|A - Q'\|_F^2 = \text{tr}(A^T A) - 2\text{tr}(A^T Q') + \text{tr}(Q'^T Q') \tag{29}$$

Since Q' is orthogonal, we have $\text{tr}(Q'^T Q') = n$. Moreover, $\text{tr}(A^T A) = \sum_{i=1}^n \sigma_i^2$ is constant with respect to Q'. Consequently, the minimization problem is equivalent to:

$$\max_{Q'\in\mathcal{O}(n)} \text{tr}(A^T Q') \tag{30}$$

Substituting the singular value decomposition of A:

$$\text{tr}(A^T Q') = \text{tr}(V\Sigma U^T Q') \tag{31}$$

By the cyclic property of the trace operator:

$$\text{tr}(A^T Q') = \text{tr}(\Sigma U^T Q' V) \tag{32}$$

Define $R = U^T Q' V$. Since U, Q', V are all orthogonal matrices, $R \in \mathcal{O}(n)$. Thus:

$$\text{tr}(A^T Q') = \text{tr}(\Sigma R) = \sum_{i=1}^n \sigma_i r_{ii} \tag{33}$$

where r_{ii} denotes the i-th diagonal element of R.

For any orthogonal matrix R, the diagonal elements satisfy $|r_{ii}| \leq 1$ for all i. Since $\sigma_i > 0$, the sum $\sum_{i=1}^n \sigma_i r_{ii}$ is maximized when $r_{ii} = 1$ for all i, which corresponds to $R = I$. The maximum value is:

$$\max_{R\in\mathcal{O}(n)} \text{tr}(\Sigma R) = \sum_{i=1}^n \sigma_i \tag{34}$$

The condition $R = I$ implies:

$$U^T Q' V = I \quad \Rightarrow \quad Q' = UV^T \tag{35}$$

This uniquely determines the optimal orthogonal approximation as $Q = UV^T = \text{msign}(A)$.

Now consider the case where A is rank-deficient with $\text{rank}(A) = r < n$. The singular value decomposition takes the form:

$$A = U\Sigma V^T \tag{36}$$

where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r, 0, \ldots, 0)$ with $\sigma_i > 0$ for $i \leq r$ and $\sigma_i = 0$ for $i > r$. The optimization problem remains:

$$\max_{Q' \in \mathcal{O}(n)} \text{tr}(A^T Q') = \max_{R \in \mathcal{O}(n)} \text{tr}(\Sigma R) = \max_{R \in \mathcal{O}(n)} \sum_{i=1}^{r} \sigma_i r_{ii} \tag{37}$$

Since only the first r singular values are nonzero, the maximum is attained when $r_{ii} = 1$ for $i = 1, \ldots, r$. The remaining diagonal elements r_{ii} for $i > r$ can take any values satisfying the orthogonality constraints of R. Therefore, the choice $R = I$ remains feasible and achieves the maximum:

$$\max_{R \in \mathcal{O}(n)} \sum_{i=1}^{r} \sigma_i r_{ii} = \sum_{i=1}^{r} \sigma_i \tag{38}$$

This corresponds to $Q' = UV^T$, which is precisely $\text{msign}(A)$ as defined for rank-deficient matrices.

In both the full-rank and rank-deficient cases, the matrix sign function achieves the minimum Frobenius distance to A over all orthogonal matrices:

$$\text{msign}(A) = \arg \min_{Q' \in \mathcal{O}(n)} \|A - Q'\|_F$$

This completes the proof.

B. Approximation of the Msign Function Using Newton-Schulz Iteration

The Newton-Schulz iteration provides an iterative method to approximate the matrix sign function, $\text{msign}(A)$, for a matrix $A \in \mathbb{R}^{m \times n}$ with singular value decomposition $A = U\Sigma V^T$. As defined, $\text{msign}(A) = UV^T$, which retains the left and right singular vectors while replacing all singular values with 1.

To apply the iteration, first normalize A to obtain $Y_0 = A/\|A\|_{\ell_2 \to \ell_2}$ (using the spectral norm, ensuring all singular values of X_0 lie in $(0, 1]$) or alternatively $Y_0 = A/\|A\|_F$ (using the Frobenius norm). Then, iterate:

$$Y_{t+1} = \frac{3}{2} Y_t - \frac{1}{2} Y_t Y_t^T Y_t. \tag{39}$$

As $t \to \infty$, $Y_t \to UV^T = \text{msign}(A)$, provided the singular values of Y_0 are in $(0, \sqrt{3})$.

This convergence arises because the iteration effectively applies the univariate cubic function $f(x) = \frac{3}{2}x - \frac{1}{2}x^3$ to each singular value of Y_t. For $0 < x < \sqrt{3}$, repeated application of $f(x)$ drives the value toward 1, approximating the sign function $\text{sign}(x)$ (which is 1 for positive x). The spectral normalization ensures the initial singular values satisfy this condition more robustly than the Frobenius norm, though the latter may suffice in practice.

More generally, the Newton-Schulz iteration belongs to a family of degree-$2n + 1$ polynomial iterations of the form

$$Y_{t+1} = aY_t + bY_t Y_t^T Y_t + c(Y_t Y_t^T)^2 Y_t + \cdots + z(Y_t Y_t^T)^n Y_t, \tag{40}$$

where coefficients $a, b, c, \ldots, z$ are chosen such that the corresponding univariate polynomial $g(x) = ax + bx^3 + cx^5 + \cdots + zx^{2n+1}$ approximates $\text{sign}(x)$. These coefficients can be tuned to optimize convergence speed. This approach is classical, dating back to works such as (Kovarik, 1970) and (Björck & Bowie, 1971), and is discussed in (Higham, 2008).

In our experiments, we adopt a quintic Newton-Schulz-style iteration of the form

$$X_{t+1} = aX_t + b\, X_t X_t^T X_t + c\, (X_t X_t^T)^2 X_t, \tag{41}$$

with empirically tuned coefficients $a = 3.4445$, $b = -4.7750$, and $c = 2.0315$. These coefficients are chosen to achieve faster per-iteration convergence at the expense of strict guaranteed convergence to $\text{msign}(A)$, trading some approximation accuracy for computational efficiency in large-scale deep learning settings.

For backpropagation, given the gradient of the loss $\mathcal{L}$ with respect to Y_k as $\partial\mathcal{L}/\partial Y_k$, gradients propagate to A via the chain rule. The backpropagation for the k-th step is:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial Y_k} =& a\frac{\partial \mathcal{L}}{\partial Y_{k+1}} + b\Big[\frac{\partial \mathcal{L}}{\partial Y_{k+1}}(Y_k^T Y_k) + Y_k\Big(\frac{\partial \mathcal{L}}{\partial Y_{k+1}}^T Y_k + Y_k^T \frac{\partial \mathcal{L}}{\partial Y_{k+1}}\Big)\Big] \\ &+ c\Big[\frac{\partial \mathcal{L}}{\partial Y_{k+1}}(Y_k^T Y_k)^2 + 2Y_k(Y_k^T Y_k)\Big(\frac{\partial \mathcal{L}}{\partial Y_{k+1}}^T Y_k + Y_k^T \frac{\partial \mathcal{L}}{\partial Y_{k+1}}\Big)\Big].\end{aligned} \tag{42}$$

C. Datasets and Experimental Settings

C.1. Datasets

Corpus of Linguistic Acceptability (CoLA) The Corpus of Linguistic Acceptability (Warstadt et al., 2019) consists of English sentences extracted from linguistic literature, annotated by linguists for grammatical acceptability. The task requires binary classification of whether a given sentence conforms to the grammatical constraints of standard English, serving as a fundamental assessment of a model's syntactic competence. Performance is evaluated using both Matthews Correlation Coefficient (MCC) and accuracy.

Stanford Sentiment Treebank (SST-2) Derived from movie reviews with manually annotated parse trees, SST-2 (Socher et al., 2013) provides a benchmark for sentiment classification at the sentence level. Each instance comprises a sentence annotated with binary sentiment polarity (positive or negative). The dataset challenges models to capture compositional sentiment semantics without utilizing structural phrase-level annotations during fine-tuning.

Microsoft Research Paraphrase Corpus (MRPC) MRPC (Dolan & Brockett, 2005) comprises sentence pairs automatically extracted from online news sources, with human annotations indicating whether the pairs are semantically equivalent (paraphrases). This binary classification task evaluates the model's ability to recognize semantic equivalence despite lexical and syntactic variation between source and target sentences.

Semantic Textual Similarity Benchmark (STS-B) Sourced from news headlines, video and image captions, and natural language inference datasets, STS-B (Cer et al., 2017) requires the prediction of continuous similarity scores ranging from 0 to 5 for sentence pairs. The task assesses the model's capacity to quantify semantic relatedness, with performance measured by Pearson and Spearman correlation coefficients against human judgments.

Quora Question Pairs (QQP) This dataset consists of question pairs derived from the Quora platform, annotated to identify duplicate questions that seek identical information. The binary classification task tests the model's ability to distinguish between semantically equivalent inquiries despite surface form variations, presenting challenges in paraphrase detection and semantic matching.

Multi-Genre Natural Language Inference (MNLI) MNLI (Williams et al., 2018) expands natural language inference across multiple genres of written and spoken English, including transcribed speech, popular fiction, and government reports. Given a premise and hypothesis sentence, the task requires three-way classification into entailment, contradiction, or neutral relationships. The dataset includes matched (in-domain) and mismatched (cross-domain) test sets to evaluate generalization capabilities.

Question-answering Natural Language Inference (QNLI) Constructed from the Stanford Question Answering Dataset (Rajpurkar et al., 2016), QNLI presents a sentence-level entailment task where premise sentences (contextual paragraphs) and hypothesis sentences (questions) are paired. The binary classification task determines whether the premise sentence contains the answer to the question, effectively converting reading comprehension into textual entailment.

Recognizing Textual Entailment (RTE) RTE aggregates data from annual textual entailment challenges (RTE1–RTE5) (Dagan et al., 2005; Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009), combining news and Wikipedia text into a binary classification benchmark. The task requires the model to identify whether a given hypothesis can be inferred from the corresponding premise, representing a fundamental evaluation of natural language inference capabilities.

C.2. Experimental Settings

Following the standard BERT paradigm, we implement Spikingformer with SmoothSpike and SpikeLM with SmoothSpike via pre-training and fine-tuning.

Pre-training Configuration We perform masked language modeling (MLM) pre-training with a masking probability of 15%. We use a global batch size of 512, distributed across 8 nodes with 64 sequences per device and a gradient accumulation step of 1. Optimization is performed using AdamW with an initial learning rate of 2×10^{-4} and no weight decay. The learning rate follows a linear decay schedule after a linear warm-up of 5,000 steps. Each sequence contains a maximum of 128 tokens. Spikingformer with SmoothSpike is trained on approximately 36.0 billion tokens, compared to 52.4 billion for SpikeLM with SmoothSpike. All datasets are accessed via the Hugging Face Datasets library: Stories[3], BookCorpus[4], CC-News[5], OpenWebText[6], and Wikipedia[7].

Fine-tuning Configuration We fine-tune the pre-trained model on individual GLUE tasks, initializing from the checkpoint. For Spikingformer with SmoothSpike, we employ a per-device batch size of 32, whereas for SpikeLM with SmoothSpike, the batch size is set to 16. The maximum sequence length is 128 tokens. We use a learning rate of 2×10^{-5} without weight decay, following a linear decay schedule without warm-up steps. Gradient accumulation is disabled.

[3] https://huggingface.co/datasets/roneneldan/TinyStories
[4] https://huggingface.co/datasets/bookcorpus
[5] https://huggingface.co/datasets/vblagoje/cc_news
[6] https://huggingface.co/datasets/Skylion007/openwebtext
[7] https://huggingface.co/datasets/wikimedia/wikipedia