ILIF: Temporal Inhibitory Leaky Integrate-and-Fire Neuron for Overactivation in Spiking Neural Networks

Kai Sun[1] , **Peibo Duan**[1*] , **Levin Kuhlmann**[1] , **Beilun Wang**[2] and **Bin Zhang**[3]
[1]Department of Data Science and AI, Monash University, Melbourne, Australia
[2]School of Computer Science and Engineering, Southeast University, China
[3]School of Software, Northeastern University, China
{kai.sun1, peibo.duan, levin.kuhlmann}@monash.edu, beilun@seu.edu.cn, zhangbin@mail.neu.edu.cn

Abstract

The Spiking Neural Network (SNN) has drawn increasing attention for its energy-efficient, event-driven processing and biological plausibility. To train SNNs via backpropagation, surrogate gradients are used to approximate the non-differentiable spike function, but they only maintain nonzero derivatives within a narrow range of membrane potentials near the firing threshold—referred to as the surrogate gradient support width γ. We identify a major challenge, termed **the dilemma of** γ: a relatively large γ leads to overactivation, characterized by excessive neuron firing, which in turn increases energy consumption, whereas a small γ causes vanishing gradients and weakens temporal dependencies. To address this, we propose a temporal Inhibitory Leaky Integrate-and-Fire (ILIF) neuron model, inspired by biological inhibitory mechanisms. This model incorporates interconnected inhibitory units for membrane potential and current, effectively mitigating overactivation while preserving gradient propagation. Theoretical analysis demonstrates ILIF's effectiveness in overcoming the γ dilemma, and extensive experiments on multiple datasets show that ILIF improves energy efficiency by reducing firing rates, stabilizes training, and enhances accuracy. The code is available at github.com/kaisun1/ILIF.

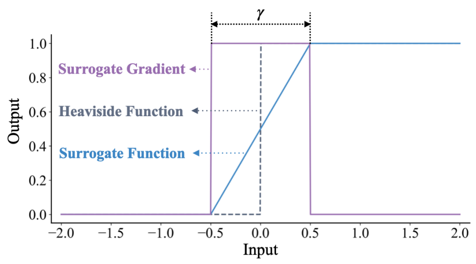
1 Introduction

The Spiking Neural Network (SNN), recognized as the third generation of neural networks, is distinguished by the simulation of neuronal message passing through spike-based activations [Maass, 1997]. Unlike the traditional Artificial Neural Network (ANN), which process information with continuous values, SNN, driven by the dynamic accumulation of membrane potential to trigger discrete activation events, effectively mimics the behavior of biological neurons by propagating discrete spike signals (0 or 1) between neurons [Tavanaei *et al.*, 2019]. This spike-based processing endows SNN with remarkable energy efficiency, especially when deployed on neuromorphic hardware platforms such as Intel's Loihi, IBM's TrueNorth, and Tianjic chips [Cai and Li, 2021].

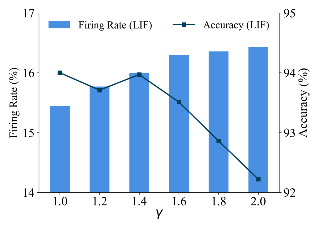
A key challenge in gradient-based optimization for training SNNs is its inherent non-differentiability due to the discrete nature of spike transmissions between neurons. To resolve this issue, surrogate gradient (SG) has been introduced to enable backpropagation in SNNs [Neftci *et al.*, 2019]. However, these approximations can cause or worsen two major issues: overactivation and gradient vanishing. Specifically, overactivation occurs when the accumulated membrane potential surpasses twice the threshold, causing excessive spiking that drives up energy consumption and masks essential temporal information. Strategies such as adaptive thresholding, residual membrane potential modulation, and normalization have been proposed to regulate neuronal activity and mitigate overactivation [Wei *et al.*, 2023; Wang and Yu, 2024; Jiang *et al.*, 2024]. Gradient vanishing, on the other hand, stems from the mismatch between surrogate gradients and discrete spikes, as well as membrane potential decay, resulting in ineffective backpropagation. Current solutions, such as residual learning and adaptive mechanisms [Fang *et al.*, 2021a; Yao *et al.*, 2022], enhance gradient propagation and improve training efficiency.

Existing research often treats overactivation and gradient vanishing as separate issues, overlooking the conflicting effects of the SG on both phenomena. As shown in Figure 1a, given the support width γ in the SG, Figure 1b demonstrates that a larger γ results in elevated firing rates and reduced accuracy (more details will be presented in Section 4.1). However, as [Huang *et al.*, 2024] points out, when the threshold remains constant, a smaller γ risks gradient vanishing. In this paper, this contradiction is referred to as **the dilemma of** γ, which highlights the need to balance the mitigation of overactivation with the preservation of gradient flow. Inspired by the brain's efficient spike regulation through feedforward and feedback inhibition, which controls excessive activation, our objective is to design a module that mitigates the γ-induced conflict between overactivation and gradient vanishing.

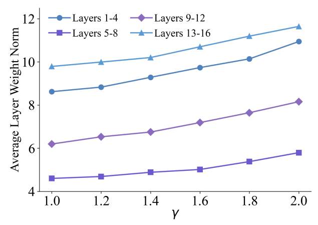
This study conducts a theoretical examination of the relationship between γ and both excessive activation and the disappearance of gradients. To address the difficulty of balancing overactivation and gradient vanishing solely by adjusting γ, we propose a temporal Inhibitory Leaky Integrate-and-Fire (ILIF) neuron model with two inhibitory units, namely the membrane potential inhibitory unit (MPIU) and the current inhibitory unit (CIU), which are designed to mimic the



(a) Surrogate gradient



(b) Firing & accuracy vs. γ



(c) Weight norm vs. γ

Figure 1: Effect of SG support width (γ) on network performance: (a) Surrogate gradient method. (b) Changes in firing rate and accuracy with γ. (c) Average layer weight norm variation with γ.

inhibitory mechanism of the human brain. The major contributions of this work are as follows:

- **Theoretical Analysis:** We conduct an in-depth mathematical investigation to clarify how the configuration of γ affects overactivation and its conflict with gradient vanishing, while analyzing how ILIF effectively mitigates both issues.
- **SNN Modeling:** The proposed MPIU and CIU in the ILIF neuron model function as feedforward and feedback inhibition, temporally interacting to reduce excessive activation and mitigate gradient vanishing.
- **Experimental Validation:** Comprehensive experiments demonstrate that the ILIF model's temporal inhibition mechanism generates fewer spikes while achieving a more stable training process and higher accuracy.

2 Related Work

2.1 Overactivation Control

To mitigate overactivation in SNN, prior works have proposed adaptive thresholds, residual potential modulation, and normalization. Adaptive threshold methods dynamically raise or lower the firing threshold based on recent activity patterns [Ding *et al.*, 2022; Wei *et al.*, 2023; Fang *et al.*, 2020; Zhang *et al.*, 2019]. Residual modulation methods, such as TC-LIF and CLIF, reduce post-spike membrane potential to suppress reactivation, mimicking the effect of afterhyperpolarization (AHP) [Wang and Yu, 2024; Niu and Wei, 2023]. Normalization techniques standardize input distributions across layers and time to stabilize spiking activity [Jiang *et al.*, 2024; Guo *et al.*, 2023; Duan *et al.*, 2022]. However, these methods often rely on handcrafted rules or instantaneous signals, lacking temporal adaptability and biological interpretability. In contrast, our ILIF model integrates inhibitory units that accumulate both short- and long-term activity, and crucially, adjusts inhibition based on the post-spike membrane potential, providing more precise, causal, and biologically aligned suppression than methods like TC-LIF and CLIF, which depend on pre-spike estimates and risk over-inhibition.

2.2 Improving Gradient Propagation

Due to the intrinsic properties of spiking neurons—namely, spike-based activation and membrane potential decay—temporal gradients often vanish over time. A key factor contributing to this vanishing is the leakage of membrane potential, which diminishes the impact of earlier inputs as time progresses. To mitigate this, models such as PLIF [Fang *et al.*, 2021b] adaptively adjust the leakage rate, preserving critical temporal information. Similarly, gating mechanisms employed in models like GLIF [Yao *et al.*, 2022], STC-LIF [Wang and Yu, 2024], and SpikGRU [Dampfhoffer *et al.*, 2022] enhance temporal dependencies by regulating information flow within neurons. However, these methods typically incur additional computational overhead and are limited to intra-neuron dynamics. In contrast, our approach introduces biologically inspired temporal connections between inhibitory units, serving as shortcuts that simultaneously facilitate backward gradient propagation and forward inhibitory signaling, thereby enhancing temporal dependencies without increasing model parameters.

3 Preliminary

3.1 Vanilla LIF Neuron Model

As shown in Figures 2b and 2c, the LIF serves as a fundamental computational framework, encompassing membrane potential integration, leakage, and spike firing upon exceeding the threshold. The model is described by the following equations:

$$\boldsymbol{U}^l[t] = \lambda \boldsymbol{m}^l[t-1] + \boldsymbol{I}^l[t] \tag{1}$$

$$\boldsymbol{I}^l[t] = \boldsymbol{W}^l \boldsymbol{S}^{l-1}[t] \tag{2}$$

$$\boldsymbol{S}^l[t] = \mathbb{H}(\boldsymbol{U}^l[t] - V_{th}) = \begin{cases} 1, & \boldsymbol{U}^l[t] \geq V_{th} \\ 0, & \text{Otherwise} \end{cases} \tag{3}$$

$$\boldsymbol{m}^l[t] = \boldsymbol{U}^l[t] - \boldsymbol{S}^l[t] V_{th} \tag{4}$$

In the LIF model, $\boldsymbol{U}^l[t]$ is the membrane potential in the l-th layer at time step t, combining the postsynaptic current $\boldsymbol{I}^l[t]$ and the residual membrane potential from the previous time step $\boldsymbol{m}^l[t-1]$ with a decay factor λ. The postsynaptic current $\boldsymbol{I}^l[t]$ is computed as the product of the synaptic weight $\boldsymbol{W}^l$ and the input spike $\boldsymbol{S}^{l-1}[t]$. The output spike $\boldsymbol{S}^l[t]$ is generated using the Heaviside function $\mathbb{H}(\cdot)$, which outputs a spike ($\boldsymbol{S}^l[t] = 1$) when the membrane potential $\boldsymbol{U}^l[t]$ exceeds the threshold V_{th} and no spike ($\boldsymbol{S}^l[t] = 0$) otherwise. After firing, the membrane potential $\boldsymbol{m}^l[t]$ is softly reset by subtracting V_{th}.

3.2 Spatio-Temporal Backpropagation

Training SNN directly involves implementing Backpropagation Through Time (BPTT) [Werbos, 1990] and using SG to

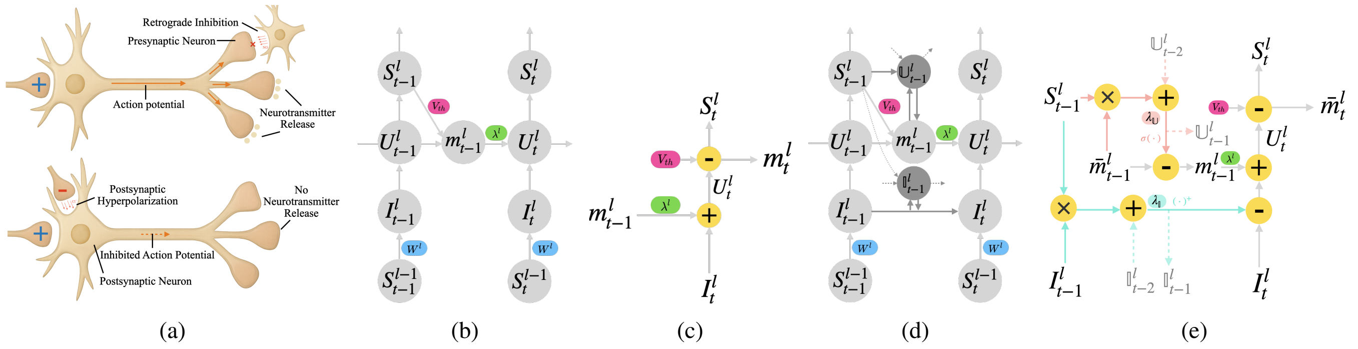


Figure 2: (a) Diagram of the inhibition mechanism. (b) Structure of the vanilla LIF model. (c) Internal operations of the vanilla LIF model. (d) Structure of the ILIF model. (e) Internal operations of the ILIF model.

handle the non-differentiability of spike signals. The gradient of the loss function $\mathcal{L}$ with respect to the weight $\boldsymbol{W}^l$ at layer l is calculated across all time steps T. It is given by

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^l} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} \cdot \frac{\partial \boldsymbol{U}^l[t]}{\partial \boldsymbol{W}^l}, l = L, L-1, \cdots, 1, \quad (5)$$

This gradient is computed by decomposing it into spatial and temporal components:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} = \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \cdot \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{\text{Spatial Term}} + \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+1]} \cdot \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{U}^l[t]}}_{\text{Temporal Term}}, \quad (6)$$

The **Spatial Term** represents the gradient contribution from the current time step, while the **Temporal Term** accounts for the influence of future time steps on the current gradient. It is calculated recursively as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \sum_{t'=t+1}^{T} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t']} \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \\ &= \sum_{t'=t}^{T} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t']} \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \end{aligned} \quad (7)$$

where the term $\epsilon^l[t]$ for LIF model is as:

$$\epsilon^l[t] \triangleq \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} \quad (8)$$

Detailed derivations are provided in Appendix A. The derivative $\frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}$ is approximated with the SG $H'(\boldsymbol{U}^l[t])$, where $H(\cdot)$ provides a smooth approximation of the Heaviside function. A common choice for the SG is the rectangular function, defined as:

$$\frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} = H'(\boldsymbol{U}^l[t]) = \frac{1}{\gamma} \mathbb{1}\left(\left| \boldsymbol{U}^l[t] - V_{\text{th}} \right| < \frac{\gamma}{2} \right), \quad (9)$$

where $\mathbb{1}(\cdot)$ is the indicator function, and γ controls the SG support width. γ is typically set to V_{th} [Meng *et al.*, 2023]. In this case

$$\epsilon^l[t] \triangleq 1 - V_{\text{th}} H'(\boldsymbol{U}^l[t]) \quad (10)$$

The gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]}$ in Eq. (7) varies depending on the layers:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} = \begin{cases} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^L[t]}, & \text{if } l = \mathcal{L}, \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^{l+1}[t]} \frac{\partial \boldsymbol{U}^{l+1}[t]}{\partial \boldsymbol{S}^l[t]}, & \text{if } l = \mathcal{L}-1, \ldots, 1. \end{cases} \quad (11)$$

where $\frac{\partial \boldsymbol{U}^{l+1}[t]}{\partial \boldsymbol{S}^l[t]} = \boldsymbol{W}^{l+1}$.

3.3 Inhibitory Mechanism

In neural systems, various inhibitory mechanisms ensure stable and efficient signal processing. For instance, after a neuron fires an action potential, its membrane potential undergoes afterhyperpolarization. This causes the potential to drop below the resting level, reducing excitability and preventing immediate re-firing. The effect of afterhyperpolarization accumulates over successive spikes, intensifying suppression over time. Additionally, activated downstream neurons send retrograde inhibitory signals to presynaptic neurons by modulating ion channel activity and suppressing neurotransmitter release. This process, mediated by GABAergic interneurons and commonly referred to as retrograde inhibition, suppresses presynaptic currents and limits excessive signal transmission [Bellec *et al.*, 2018; Zenke and Ganguli, 2018]. Inspired by these mechanisms, we enhance the LIF model by incorporating inhibitory processes to reduce overactivation and facilitate more effective gradient flow. The corresponding inhibitory mechanisms are illustrated in Figure 2a.

4 Methodology

4.1 The Dilemma of γ

The dilemma of γ refers to the challenge of balancing overactivation and gradient vanishing in neural networks. The correlation between γ and overactivation has been underexplored, motivating our investigation into this correlation.

Lemma 1. *The likelihood of experiencing overactivation is positively correlated with* γ.

Proof. When the neural network converges to an optimum such that no further weight updates occur, we consider the system to have reached an equilibrium. Let $\boldsymbol{W}_1$ be the equilibrium weights when $\gamma = \gamma_1$ and $\boldsymbol{W}_2$ be the equilibrium weights when $\gamma = \gamma_2$, with $\gamma_2 > \gamma_1$. Suppose that there are neurons in the firing state (requiring $\boldsymbol{W}^l \boldsymbol{S}^{l-1}[t] > 0$ to accumulate sufficient membrane potential), whereas others remain inactive (requiring $\boldsymbol{W}^l \boldsymbol{S}^{l-1}[t] < 0$ to suppress activation).

In the output layer, the gradient terms related to the loss function are: $\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^L[t]} = \boldsymbol{Y}^L[t] - \hat{\boldsymbol{Y}}^L[t]$, $\frac{\partial \boldsymbol{U}^L[t]}{\partial \boldsymbol{W}^L} = \boldsymbol{S}^{L-1}[t]$, where $\boldsymbol{Y}^L[t]$ represents the average output (within [0,1]), and $\hat{\boldsymbol{Y}}^L[t]$ represents the predicted values encoded in a one-hot format (taking values of either 0 or 1). As derived from Equations (5), (7), (9) and (10), the expected gradient with respect to W^L is

$$\frac{\partial \mathcal{L}}{\partial W^L} = \sum_{t'=t}^{T} \left(\overbrace{(\boldsymbol{Y}^L[t'] - \hat{\boldsymbol{Y}}[t'])}^{[0,1]} \overbrace{H'(U^L[t'])}^{\geq 0} \prod_{t''=t+1}^{t'} \lambda \overbrace{\left(1 - V_{\text{th}} H'(U^L[t''])\right)}^{\geq 0} \right) \overbrace{\boldsymbol{S}^{L-1}[t]}^{\geq 0} \begin{cases} \leq 0, & \hat{\boldsymbol{Y}}[t'] = 1 \\ \geq 0, & \hat{\boldsymbol{Y}}[t'] = 0 \end{cases} \tag{12}$$

Under the SG framework, the gradient for a given sample becomes zero whenever the membrane potential moves outside the range $\left[V_{\text{th}} - \frac{\gamma}{2}, V_{\text{th}} + \frac{\gamma}{2}\right]$. Neurons whose membrane potentials exceed or fall below this interval stop receiving weight updates. As λ increases from λ_1 to λ_2, neurons previously outside the range may re-enter it, allowing further updates during the transition from $\boldsymbol{W}_1$ to $\boldsymbol{W}_2$.

According to Equation (12), Neurons expected to fire ($\hat{\boldsymbol{Y}}[t'] = 1$) receive negative gradients, which increase positive weights. Conversely, neurons expected to remain inactive ($\hat{\boldsymbol{Y}}[t'] = 0$) receive positive gradients, which decrease negative weights. This process contributes to an overall increase in the weight norm $\|\boldsymbol{W}\|$. Additionally, based on Equation (11) and the chain rule, changes in the final layer's weights propagate to earlier layers, causing their weights to increase or decrease accordingly.

During activation, the membrane potential increment is $\boldsymbol{W}^l \boldsymbol{S}^{l-1}[t]$. A larger $\|\boldsymbol{W}\|$ amplifies this increment, increasing the chance of exceeding twice the threshold and causing overactivation. As shown in Figure 1c, increasing γ leads to a higher average layer weight norm, which amplifies the membrane potential further, thereby heightening the risk of overactivation. $\square$

Lemma 2. *The likelihood of experiencing gradient vanishing is inversely correlated with γ.*

The gradient in Eq. (7) includes the product $\prod \epsilon^l[t]$, where $\epsilon^l[t] = 1 - V_{\text{th}}/\gamma$ when the membrane potential is near threshold and $\gamma > V_{\text{th}}$. As γ decreases toward V_{th}, this factor becomes smaller, reducing the gradient magnitude over time. At $\gamma = V_{\text{th}}$, $\epsilon^l[t] = 0$, and gradients are completely blocked. See Appendix B for details.

As seen from Lemmas 1 and 2, the impact of γ is twofold: increasing γ exacerbates overactivation, while decreasing it increases the risk of gradient vanishing. This trade-off demonstrates the limitation of adjusting γ alone to improve SNN performance. Thus, specialized modules are needed to strike an optimal balance between these competing effects.

4.2 ILIF Neuron Model

To address overactivation and gradient vanishing in the vanilla LIF model, we propose the ILIF model with two biologically-inspired inhibitory mechanisms. The MPIU provides long-term inhibition by mimicking afterhyperpolarization, stabilizing neuronal excitability. The CIU delivers short-term inhibition via retrograde-like feedback, regulating presynaptic currents. The ILIF structure and internal operations are illustrated in Figures 2d and 2e.

MPIU Each spike contributes to the integration of the post-spike membrane potential $\bar{\boldsymbol{m}}^l[t]$ into $\mathbb{U}^l[t]$, which decays slowly ($\lambda_{\mathbb{U}} \approx 1$) to maintain a long-term memory of spiking activity:

$$\mathbb{U}^l[t] = \lambda_{\mathbb{U}} \left(\mathbb{U}^l[t-1] + \boldsymbol{S}^l[t] \cdot \bar{\boldsymbol{m}}^l[t] \right) \tag{13}$$

This unit accumulates historical firing patterns, retaining the neuron's firing history and mimicking the afterhyperpolarization phenomenon in biological systems. As firing activity accumulates, the accumulated inhibition $\mathbb{U}^l[t]$ grows, progressively reducing excitability and preventing excessive firing. This inhibition directly affects the membrane potential before subsequent firings:

$$\boldsymbol{m}^l[t] = \bar{\boldsymbol{m}}^l[t] - \boldsymbol{S}^l[t] \cdot \sigma\left(\mathbb{U}^l[t]\right) \tag{14}$$

where $\sigma(\cdot)$ is a bounded sigmoid function that allows inhibition to increase with historical firing, in contrast to linear mappings or unbounded nonlinear functions that may lead to instability. This mirrors synaptic plasticity mechanisms that adjust synaptic strengths based on past activity, preventing excessive suppression and maintaining learnability over extended timescales. Moreover, the temporal linkage within MPIU mitigates vanishing gradients by offering a direct pathway for backpropagation, as further analyzed in Section 4.3.

CIU Complementing the MPIU's long-term inhibition, the CIU provides rapid, short-term feedback. With a decay coefficient $\lambda_{\mathbb{I}}$ close to 0, primarily inhibiting the incoming current based on the previous time step's current and the current spikes from the next layer:

$$\mathbb{I}^l[t] = \lambda_{\mathbb{I}} \left(\mathbb{I}^l[t-1] + \boldsymbol{S}^l[t] \cdot \boldsymbol{I}^l[t] \right) \tag{15}$$

The incoming current is adjusted as follows:

$$\boldsymbol{I}^l[t] = \boldsymbol{S}^{l-1}[t] \cdot W^l - \mathbb{I}^l[t-1]^+ \tag{16}$$

Here, $\mathbb{I}^l[t-1]^+$ ensures non-negative inhibition, mimicking the rapid feedback mediated by GABAergic interneurons in biological systems. By delivering rapid inhibition tied to the prior spikes and current, the CIU stabilizes excitability in tandem with MPIU.

Combining these mechanisms, the ILIF model evolves according to the following equations, with the corresponding pseudocode provided in Appendix E:

$$\begin{aligned}
\mathbb{I}^l[t-1] &= \lambda_{\mathbb{I}}\left(\mathbb{I}^l[t-2] + \boldsymbol{S}^l[t-1]\cdot \boldsymbol{I}^l[t-1]\right)\\
\boldsymbol{I}^l[t] &= \boldsymbol{S}^{l-1}[t]\cdot W^l - \mathbb{I}^l[t-1]^+\\
\boldsymbol{U}^l[t] &= \lambda^l \boldsymbol{m}^l[t-1] + \boldsymbol{I}^l[t]\\
\boldsymbol{S}^l[t] &= \mathbb{H}\left(\boldsymbol{U}^l[t] - V_{\text{th}}\right) = \begin{cases}1, & \boldsymbol{U}^l[t]\geq V_{\text{th}}\\ 0, & \text{otherwise}\end{cases} \qquad (17)\\
\bar{\boldsymbol{m}}^l[t] &= \boldsymbol{U}^l[t] - \boldsymbol{S}^l[t]V_{\text{th}}\\
\mathbb{U}^l[t] &= \lambda_{\mathbb{U}}\left(\mathbb{U}^l[t-1] + \boldsymbol{S}^l[t]\cdot\bar{\boldsymbol{m}}^l[t]\right)\\
\boldsymbol{m}^l[t] &= \bar{\boldsymbol{m}}^l[t] - \boldsymbol{S}^l[t]\cdot\sigma\left(\mathbb{U}^l[t]\right)
\end{aligned}$$

4.3 Analysis of the ILIF Model

The ILIF model builds directly upon the vanilla LIF model by introducing inhibitory decay factors $\lambda_{\mathbb{U}}$ and $\lambda_{\mathbb{I}}$. Setting these factors to zero removes all inhibitory effects, reducing the ILIF model to the standard LIF model. This demonstrates that the ILIF model's inhibitory mechanisms are natural extensions that refine the neuron's behavior without altering its fundamental structure. In this analysis, we explore the relationship between the ILIF and LIF models and the underlying principles that enable the ILIF model to overcome the limitations of the LIF model.

Theorem 1. *$\boldsymbol{W}'$ is the equilibrium weight in the vanilla LIF, and $\boldsymbol{W}''$ is the new equilibrium weight in the ILIF, $\|\boldsymbol{W}''\| > \|\boldsymbol{W}'\|$.*

Proof. In the vanilla LIF model, the gradient of the loss $\mathcal{L}$ with respect to $\boldsymbol{U}^l[t]$ consists of both a spatial and a temporal term (denoted as $a(\boldsymbol{W})$, highlighted in blue). In contrast, the ILIF model introduces additional gradient terms (denoted as $b(\boldsymbol{W})$, highlighted in red), as shown in Equation (18). A detailed derivation is provided in Appendix C.

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial\boldsymbol{U}^l[t]} &= \frac{\partial\mathcal{L}}{\partial\boldsymbol{S}^l[t]}\frac{\partial\boldsymbol{S}^l[t]}{\partial\boldsymbol{U}^l[t]}\\
&+\sum_{t'=t+1}^{T}\left(\frac{\partial\mathcal{L}}{\partial\boldsymbol{S}^l[t']}\frac{\partial\boldsymbol{S}^l[t']}{\partial\boldsymbol{U}^l[t']}\prod_{t''=t+1}^{t'}\lambda\epsilon^l[t''-1]\right)\\
&+\sum_{t'=t+1}^{T}\left(\phi^l[t']\prod_{t''=t+1}^{t'}\lambda\epsilon^l[t''-1]\right)+\phi^l[t]
\end{aligned} \qquad (18)$$

where $\phi^l[t]$ encapsulates the derivatives associated with MPIU and CIU:

$$\begin{aligned}
\phi^l[t] = \frac{\partial\mathcal{L}}{\partial\mathbb{U}^l[t]}\left(\frac{\partial\mathbb{U}^l[t]}{\partial\bar{\boldsymbol{m}}^l[t]}\epsilon^l[t] + \frac{\partial\mathbb{U}^l[t]}{\partial\boldsymbol{S}^l[t]}\frac{\partial\boldsymbol{S}^l[t]}{\partial\boldsymbol{U}^l[t]}\right)\\
+\frac{\partial\mathcal{L}}{\partial\mathbb{I}^l[t]}\left(\frac{\partial\mathbb{I}^l[t]}{\partial\boldsymbol{S}^l[t]}\frac{\partial\boldsymbol{S}^l[t]}{\partial\boldsymbol{U}^l[t]}\right)
\end{aligned} \qquad (19)$$

Consider the vanilla LIF model without inhibition, where the equilibrium weight $\boldsymbol{W}'$ satisfies $a(\boldsymbol{W}') = 0$. The function $a(\boldsymbol{W})$ includes terms that are products of $\boldsymbol{W}$ and typically oppose the sign of $\boldsymbol{W}$. Introducing an additional gradient $b(\boldsymbol{W})$, which shares the same sign as $\boldsymbol{W}$ (see Appendix D), modifies the total gradient to $a(\boldsymbol{W}) + b(\boldsymbol{W})$, and the new equilibrium $\boldsymbol{W}''$ must satisfy $a(\boldsymbol{W}'') + b(\boldsymbol{W}'') = 0$.

If $\boldsymbol{W}'' > 0$, then $b(\boldsymbol{W}'') > 0$ and thus $a(\boldsymbol{W}'') < 0$ to maintain equilibrium. Since $a(\boldsymbol{W})$ becomes more negative as $\boldsymbol{W}$ increases, this implies $\boldsymbol{W}'' > \boldsymbol{W}'$. Similarly, if $\boldsymbol{W}'' < 0$, then $b(\boldsymbol{W}'') < 0$ and $a(\boldsymbol{W}'') > 0$, and because $a(\boldsymbol{W})$ becomes more positive as $\boldsymbol{W}$ decreases, it follows that $\boldsymbol{W}'' < \boldsymbol{W}'$.

Therefore, in both cases, the magnitude of the new equilibrium increases: $\|\boldsymbol{W}''\| > \|\boldsymbol{W}'\|$. Empirical results supporting this conclusion are provided in Section 5.3. ☐

Theorem 2. *r' is the equilibrium firing rate in the vanilla LIF, and r'' is the new equilibrium firing rate in the ILIF. It satisfies that $r'' < r'$ even $\|\boldsymbol{W}''\| > \|\boldsymbol{W}'\|$.*

Proof. According to Lemma 1, we define the excitatory input function $F(\|\mathbf{W}\|)$, which increases with $\|\mathbf{W}\|$, indicating that larger weight magnitudes result in higher excitatory input necessary for neuron firing, and the response function $R(x)$, which maps input x to firing rate r, reflecting that higher inputs lead to increased activation. Additionally, based on Equations (13) and (15), we define the inhibition function $J(r)$, which increases with the firing rate r and satisfies $J(r) \geq 0$ for all $r \geq 0$, ensuring that higher firing rates induce stronger inhibitory effects.

In the vanilla LIF without inhibition, the equilibrium firing rate is $r' = R(F(\|\boldsymbol{W}'\|))$. When inhibition is introduced, the new equilibrium rate satisfies $r'' = R(F(\|\boldsymbol{W}''\|) - J(r''))$. By Theorem 1, we know that $\|\boldsymbol{W}''\| > \|\boldsymbol{W}'\|$.

To prove $r'' < r'$, suppose instead that $r'' \geq r'$. Since R is increasing, we have

$$F(\|\boldsymbol{W}''\|) - J(r'') \geq F(\|\boldsymbol{W}'\|), \qquad (20)$$

which implies $F(\|\boldsymbol{W}''\|) - F(\|\boldsymbol{W}'\|) \geq J(r'') > 0$. Define the function $H(r) = R(F(\|\boldsymbol{W}''\|) - J(r))$. Since both R and J are strictly increasing, we have

$$\frac{dH}{dr} = R'\big(F(\|\boldsymbol{W}''\|) - J(r)\big)\left(-J'(r)\right) < 0, \qquad (21)$$

so $H(r)$ is strictly decreasing. Note that $J(r') = 0$, so

$$H(r') = R(F(\|\boldsymbol{W}''\|)) > R(F(\|\boldsymbol{W}'\|)) = r'. \qquad (22)$$

By the fixed-point theorem[1], since $H(r') > r'$ and H is decreasing, the fixed point $r'' = H(r'')$ must satisfy $r'' < r'$, contradicting our assumption.

Therefore, $r'' < r'$. This shows that even though $\|\boldsymbol{W}''\| > \|\boldsymbol{W}'\|$, the resulting firing rate is strictly lower, meaning that inhibition effectively balances the excitatory input and prevents overactivation. ☐

Theorem 3. *ILIF introduces additional gradient propagation pathways, mitigating the vanishing gradient problem.*

[1]Let $f(x)$ be strictly decreasing. If $f(x_1) > x_1$, then any fixed point x^* such that $f(x^*) = x^*$ must satisfy $x^* < x_1$.

Proof. According to Lemma 2, the vanishing gradient problem in LIF is caused by the term $\epsilon^l[t]$, which gradually accumulates to zero with lower γ. In ILIF, however, membrane potential inhibitory units introduce a direct shortcut that connects all time steps. Because each inhibitory unit is interconnected and the decay coefficient is set to 1 in our experiments, there exists a shortcut in the backward pass given by $\frac{\partial \mathbb{U}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}$ enabling each time step's inhibitory unit to transmit the gradient directly to the preceding time step. Consequently, as inhibition stops after T with the output produced, the complete chain of partial derivatives from time step $T-1$ back to t can be written as:

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[T-1]} \frac{\partial \mathbb{U}^l[T-1]}{\partial \mathbb{U}^l[T-2]} \cdots \frac{\partial \mathbb{U}^l[t+1]}{\partial \mathbb{U}^l[t]} \frac{\partial \mathbb{U}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} \\ &= \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[T-1]} \frac{\partial \mathbb{U}^l[T-1]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} \end{aligned} \tag{23}$$

Since there is no decay in these shortcuts, the gradient can flow from the time step $T-1$ back to any earlier step without attenuation. This mechanism mitigates the vanishing gradient problem and facilitates effective gradient propagation. □

5 Experiments

5.1 Experimental Settings

In this research, we conduct experiments on two types of datasets without data augmentation: standard image classification datasets (CIFAR10 and CIFAR100) and neuromorphic datasets (DVSCIFAR10 and DVSGesture). For a comprehensive and unbiased evaluation, the proposed ILIF model is implemented under various settings, including different network architectures, neuron models, and training time steps. Further details are provided in Appendix F.

5.2 Evaluation of Accuracy

As shown in Table 1, our ILIF model consistently achieves superior top-1 accuracy compared to existing methods. Specifically, our method achieves 95.49% on CIFAR10 and 78.51% on CIFAR100 using the ResNet-18 architecture with 6 time-steps. For the neuromorphic dataset, ILIF achieves an accuracy of 78.60% on DVSCIFAR10 and matches the top-1 accuracy of 97.92% achieved by other models on DVSGesture, further demonstrating its effectiveness. More experiments are provided in Appendix J.

5.3 Effectiveness of Overactivation Inhibition

We quantified the proportion of neurons activated for more than half the time. As shown in Figure 3, the ILIF model consistently exhibits lower firing rates and fewer instances of continuous activation across all layers compared to LIF model on DVSCIFAR10 and DVSGesture. Furthermore, the reduction in continuous activation rates leads to a lower overall average firing rate, especially with the activation rate by over 30% on DVSGesture. Additional figures for all datasets are provided in the Appendix G.

To better understand the firing patterns across time steps and layers, we compared the spike counts of the LIF and ILIF

Table 1: Comparisons with other SNN neuron models on CIFAR10, CIFAR100, DVSCIFAR10 and DVSGesture

Dataset	Method	Network Architecture	Time Step	Accuracy (%)
CIFAR10	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	4 / 6	92.92 / 93.16
	Dspike [Li *et al.*, 2021]	Modified ResNet-18	4 / 6	93.66 / 94.05
	GLIF [Yao *et al.*, 2022]	ResNet-18	4	94.67
	SML [Deng *et al.*, 2023]	ResNet-18	6	95.12
	CLIF [Huang *et al.*, 2024]	ResNet-18	4 / 6	94.89 / 95.41
	Ours	ResNet-18	4 / 6	**95.24 / 95.49**
CIFAR100	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	4 / 6	70.86 / 71.12
	Dspike [Li *et al.*, 2021]	Modified ResNet-18	4 / 6	73.35 / 74.24
	GLIF [Yao *et al.*, 2022]	ResNet-18	4 / 6	76.42 / 77.28
	SML [Deng *et al.*, 2023]	ResNet-18	6	78.00
	CLIF [Huang *et al.*, 2024]	ResNet-18	4 / 6	77.00 / 78.36
	Ours	ResNet-18	4 / 6	**77.43 / 78.51**
DVSCIFAR10	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	10	67.8
	Dspike [Li *et al.*, 2021]	ResNet-18	10	75.40
	OTTT [Xiao *et al.*, 2022]	VGG-11	10	76.27
	SLTT [Meng *et al.*, 2023]	VGG-11	10	77.17
	Ours	VGG-11	10	**78.60**
DVSGesture	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-17	40	96.87
	OTTT [Xiao *et al.*, 2022]	VGG-11	20	96.88
	SLTT [Meng *et al.*, 2023]	VGG-11	20	97.92
	CLIF [Huang *et al.*, 2024]	VGG-11	20	97.92
	Ours	VGG-11	20	**97.92**

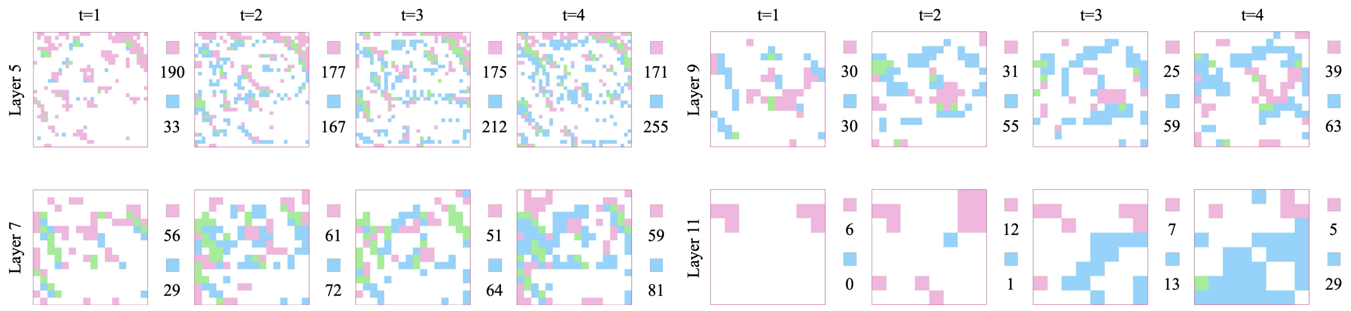
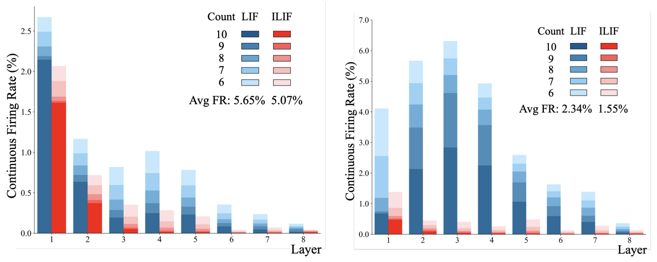


Figure 4: Comparison of firing rates between LIF and ILIF in Layers 5, 7, 9, and 11 on CIFAR10. Blue pixels indicate LIF spikes, red pixels indicate ILIF spikes, and green pixels indicate simultaneous spikes. Spike counts are displayed beside each time step.

models. According to Theorems 1 and 2, inhibitory units lead to larger synaptic weights but also induce strong inhibition. As shown in Figure 4, the ILIF model exhibits a higher firing rate in the first time step due to the absence of immediate inhibition and larger synaptic weights. This efficiently captures critical information early, avoiding prolonged accumulation and delayed activation as observed in the LIF model, where the number of activations rises rapidly across time steps. In later time steps, firing rates decrease as inhibition suppresses redundant activations, ensuring efficient information transfer.



(a) DVSCIFAR10 (b) DVSGesture

Figure 3: Continuous firing rate comparison on neuromorphic data.

5.4 Evaluation of the Effect of γ on ILIF

Figure 5 illustrates the impact of varying γ on the firing rate and accuracy of the LIF and ILIF models. As γ increases, the LIF model experiences a significant decline in accuracy due to heightened activation, accompanied by a relatively higher firing rate. In contrast, the ILIF model consistently demonstrates a lower firing rate, highlighting its inhibitory effect. Furthermore, the ILIF model works synergistically with the backpropagation shortcut provided by its inhibitory units, ensuring more stable accuracy across all γ values.

5.5 Ablation Study

We performed ablation experiments to evaluate the contributions of each inhibitory unit in the ILIF model. As shown in Table 2, removing either unit results in a noticeable drop in accuracy, confirming their necessity. To explore the generalizability of our proposed inhibitory mechanism, we integrated it into the PLIF model [Fang *et al.*, 2021b], a LIF variation that adjusts decay parameters. This integration (Table 3) significantly enhances performance, demonstrating the value of

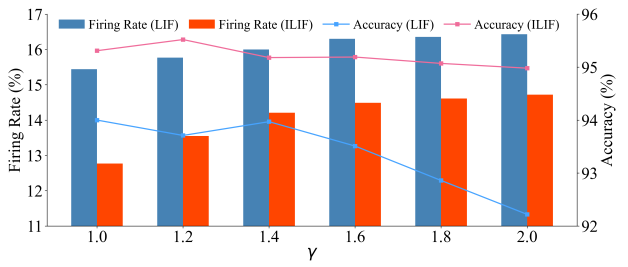


Figure 5: Firing rate and accuracy comparison for LIF and ILIF with respect to γ on CIFAR10

our inhibitory mechanism given that PLIF only modifies decay parameters and inherits the limitations of LIF neurons.

Table 2: Ablation study of ILIF model components.

Settings		CIFAR10	CIFAR100	DVSCIFAR10	DVSGesture
MPIU	CIU				
✓	✓	**95.49**	**78.51**	**78.60**	**97.92**
✓	✗	95.11	78.06	78.21	97.56
✗	✓	94.63	76.84	77.13	97.22
✗	✗	93.76	75.46	75.50	96.88

Table 3: Impact of adding inhibitory unit to PLIF model.

Method	CIFAR10	CIFAR100	DVSCIFAR10	DVSGesture
PLIF	94.25	75.65	75.32	95.49
IPLIF	95.61	78.66	78.41	96.55

6 Conclusion

This paper analyzes the limitations of the LIF model, specifically overactivation and gradient vanishing caused by the SG support width γ. Inspired by biological inhibitory mechanisms, we propose the ILIF model, which integrates MPIU and CIU to regulate neural activity and enhance gradient propagation simultaneously. Experiments show that ILIF achieves state-of-the-art performance, reduces continuous firing rates, and enhances result stability, providing an efficient and reliable solution for advancing neuromorphic computing.

A Gradient Derivation Based on LIF model

With surrogate gradient method, the gradients are computed by BPTT, meaning backpropagating the loss $\mathcal{L}$ through all time steps. The gradients at l-th are calculates as:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^l} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} \cdot \frac{\partial \boldsymbol{U}^l[t]}{\partial \boldsymbol{W}^l}, l = L, L-1, \cdots, 1, \tag{24}$$

According to the chain rule, when $t = T$, we have

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[T]} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[T]} \cdot \frac{\partial \boldsymbol{S}^l[T]}{\partial \boldsymbol{U}^l[T]}, \tag{25}$$

When $t = T-1, \dots, 1$, this gradient can be decomposed into spatial and temporal components:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} = \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \cdot \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{\text{Spatial Term}} + \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+1]} \cdot \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{U}^l[t]}}_{\text{Temporal Term}}, \tag{26}$$

And it can be further derived recursively as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+1]} \overbrace{\overbrace{\frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]}}^{\text{decay rate } \lambda} \overbrace{\left(\frac{\partial \boldsymbol{m}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \boldsymbol{m}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}\right)}^{\epsilon^l[t]}} \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+2]} \lambda \epsilon^l[t+1] \right) \lambda \epsilon^l[t] \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \cdot \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \cdot \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \cdot \lambda \epsilon^l[t] + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+2]} \cdot \lambda \epsilon^l[t+1] \cdot \lambda \epsilon^l[t] \\
&= \dots \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \cdot \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \sum_{t'=t+1}^{T} \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \cdot \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t']} \cdot \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \right) \\
&= \sum_{t'=t}^{T} \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \cdot \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t]} \cdot \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \right),
\end{aligned} \tag{27}$$

The gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]}$ varies depending on the layer, and can be expressed as:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} = \begin{cases} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^L[t]}, & \text{if } l = \mathcal{L}, \\ \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^{l+1}[t]} \frac{\partial \boldsymbol{U}^{l+1}[t]}{\partial \boldsymbol{S}^l[t]}, & \text{if } l = \mathcal{L}-1, \dots, 1. \end{cases} \tag{28}$$

B Analysis of γ's Impact on Temporal Gradient Vanishing

Here, we analyze the impact of γ on temporal gradient vanishing to complete the proof of Lemma 2 according to the analysis in [Huang *et al.*, 2024]:

Lemma 2. *The likelihood of experiencing gradient vanishing is inversely correlated with γ.*

Proof. From Equation (27), the gradient backpropagated from time t' to an earlier time t can be written as

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \cdot \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t']} \cdot \prod_{t''=t+1}^{t'} \lambda \, \epsilon^l[t''-1], \tag{29}$$

where

$$\epsilon^l[t] = 1 - V_{\text{th}} \, H'(x) = \begin{cases} 1 - \frac{V_{\text{th}}}{\gamma}, & \text{if } \left|U^l[t] - V_{\text{th}}\right| \le \frac{\gamma}{2}, \\ 0, & \text{if } \left|U^l[t] - V_{\text{th}}\right| > \frac{\gamma}{2}. \end{cases} \tag{30}$$

The key observation is that $\epsilon^l[t]$ can become very small or even zero, depending on γ. To avoid spatial gradient explosion, prior work [Wu *et al.*, 2019] often sets $\gamma \geq V_{\text{th}}$. In this regime ($\gamma > V_{\text{th}}$), we have

$$0 < \frac{V_{\text{th}}}{\gamma} < 1 \quad \Longrightarrow \quad 0 < 1 - \frac{V_{\text{th}}}{\gamma} < 1. \tag{31}$$

Hence as γ decreases from values $\gamma > V_{\text{th}}$ down toward $\gamma = V_{\text{th}}$, the quantity $\left(1 - \frac{V_{\text{th}}}{\gamma}\right)$ shrinks toward zero. Because this factor appears repeatedly in the product $\prod_{t''=t+1}^{t'} \epsilon^l[t''-1]$, it accelerates the convergence of gradients to zero (i.e., gradient vanishing) whenever a neuron is near threshold.

Furthermore, if one adopts the often-used default $\gamma = V_{\text{th}} = 1$ [Deng *et al.*, 2022a], then

$$1 - \frac{V_{\text{th}}}{\gamma} = 1 - 1 = 0, \tag{32}$$

which yields a complete cutoff. Specifically, whenever $\left|U^l[t] - V_{\text{th}}\right| \leq \frac{\gamma}{2}$, we have $\epsilon^l[t] = 0$, causing all past gradients from times $t' \leq t$ to be nullified by the product.

Therefore, as γ approaches V_{th}, the factor $1 - \frac{V_{\text{th}}}{\gamma}$ becomes vanishingly small. This markedly increases the likelihood of gradient vanishing for near-threshold events—demonstrating that **reducing** the surrogate gradient support width γ **exacerbates** vanishing of the temporal gradient, especially at the critical point $\gamma = V_{\text{th}}$. This completes the proof of Lemma 2. □

C Gradient Derivation Based on ILIF model

According to the chain rule and the propagation process illustrated in Figure 2d and 2e, we can derive the following expression:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} = \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{\text{Spatial part}} + \underbrace{\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+1]} \overbrace{\frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]}}^{\lambda} \overbrace{\frac{\partial \boldsymbol{m}^l[t]}{\partial \bar{\boldsymbol{m}}^l[t]}}^{1} \epsilon^l[t]}_{\text{Temporal part}} + \underbrace{\overbrace{\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} \xi^l[t] + \frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} \delta^l[t]}^{\text{define as } \phi^l[t]}}_{\text{Inhibitory additional part}} \tag{33}$$

In this equation,

$$\epsilon^l[t] \equiv \frac{\partial \bar{\boldsymbol{m}}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \bar{\boldsymbol{m}}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} = 1 - \frac{V_{\text{th}}}{\gamma} \geq 0 \tag{34}$$

$$\xi^l[t] \equiv \underbrace{\frac{\partial \mathbb{U}^l[t]}{\partial \bar{\boldsymbol{m}}^l[t]}}_{\geq 0} \underbrace{\epsilon^l[t]}_{\geq 0} + \underbrace{\frac{\partial \mathbb{U}^l[t]}{\partial \boldsymbol{S}^l[t]}}_{>0} \underbrace{\frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{>0} > 0 \tag{35}$$

$$\delta^l[t] \equiv \underbrace{\frac{\partial \mathbb{I}^l[t]}{\partial \boldsymbol{S}^l[t]}}_{>0} \underbrace{\frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{>0} > 0 \tag{36}$$

Next, we recursively expand the equation in a concise manner and examine the intermediate components in Appendix D.

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+1]} \lambda \epsilon^l[t] + \phi^l[t] \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+2]} \lambda \epsilon^l[t+1] + \phi^l[t+1] \right) \lambda \epsilon^l[t] + \phi^l[t] \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \lambda \epsilon^l[t] + \frac{\partial \mathcal{L}}{\partial \boldsymbol{U}^l[t+2]} \lambda \epsilon^l[t+1] \lambda \epsilon^l[t] + \phi^l[t+1] \lambda \epsilon^l[t] + \phi^l[t] \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \lambda \epsilon^l[t] + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+2]} \frac{\partial \boldsymbol{S}^l[t+2]}{\partial \boldsymbol{U}^l[t+2]} \lambda \epsilon^l[t+1] \lambda \epsilon^l[t] \\
&\quad + \phi^l[t+2] \lambda \epsilon^l[t+1] \lambda \epsilon^l[t] + \phi^l[t+1] \lambda \epsilon^l[t] + \phi^l[t] \\
&= \ldots \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} + \sum_{t'=t+1}^{T} \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t']} \frac{\partial \boldsymbol{S}^l[t']}{\partial \boldsymbol{U}^l[t']} \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \right) + \sum_{t'=t+1}^{T} \left(\phi^l[t'] \prod_{t''=t+1}^{t'} \lambda \epsilon^l[t''-1] \right) + \phi^l[t]
\end{aligned} \tag{37}$$

where the blue part is the same as the computation in vanilla LIF model while the red part is the additional gradient introduced by the inhibitory units.

D Analysis of Additional Gradient Propagation of ILIF

The ILIF model introduces an inhibitory mechanism that influences gradient propagation through time. To understand its effect, we analyze the gradient contributions across different time steps.

When $t = T$, the additional gradient is zero because the inhibition at the final time step occurs after the spike is generated, thus having no effect on the output.

When $t = T - 1$, the gradient contribution involves both the membrane potential and the input current, which can be expressed as:

$$\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[T-1]} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[T]} \frac{\partial \boldsymbol{S}^l[T]}{\partial \boldsymbol{U}^l[T]} \frac{\partial \boldsymbol{U}^l[T]}{\partial \boldsymbol{m}^l[T-1]} \frac{\partial \boldsymbol{m}^l[T-1]}{\partial \mathbb{U}^l[T-1]} \tag{38}$$

and

$$\frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[T-1]} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[T]} \frac{\partial \boldsymbol{S}^l[T]}{\partial \boldsymbol{U}^l[T]} \frac{\partial \boldsymbol{U}^l[T]}{\partial \boldsymbol{I}^l[T]} \frac{\partial \boldsymbol{I}^l[T]}{\partial \mathbb{I}^l[T-1]} \tag{39}$$

When $t = T - 2, \ldots, 1$, we recursively expand the additional gradient parts. For the membrane potential, this expansion is written as:

$$\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t+1]} \left(\frac{\partial \mathbb{U}^l[t+1]}{\partial \mathbb{U}^l[t]} + \xi^l[t+1] \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} \right) \tag{40}$$

We further expand:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t+1]} \overbrace{\left(\frac{\partial \mathbb{U}^l[t+1]}{\partial \mathbb{U}^l[t]} + \xi^l[t+1] \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} \right)}^{\text{define as } \theta^l[t]} \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} + \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+2]} \frac{\partial \boldsymbol{S}^l[t+2]}{\partial \boldsymbol{U}^l[t+2]} \frac{\partial \boldsymbol{U}^l[t+2]}{\partial \boldsymbol{m}^l[t+1]} \frac{\partial \boldsymbol{m}^l[t+1]}{\partial \mathbb{U}^l[t+1]} + \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t+2]} \theta^l[t+1] \right) \theta^l[t] \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+2]} \frac{\partial \boldsymbol{S}^l[t+2]}{\partial \boldsymbol{U}^l[t+2]} \frac{\partial \boldsymbol{U}^l[t+2]}{\partial \boldsymbol{m}^l[t+1]} \frac{\partial \boldsymbol{m}^l[t+1]}{\partial \mathbb{U}^l[t+1]} \theta^l[t] \\
&\quad + \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+3]} \frac{\partial \boldsymbol{S}^l[t+3]}{\partial \boldsymbol{U}^l[t+3]} \frac{\partial \boldsymbol{U}^l[t+3]}{\partial \boldsymbol{m}^l[t+2]} \frac{\partial \boldsymbol{m}^l[t+2]}{\partial \mathbb{U}^l[t+2]} \theta^l[t+1] \theta^l[t] + \ldots \\
&= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{m}^l[t]} \frac{\partial \boldsymbol{m}^l[t]}{\partial \mathbb{U}^l[t]} + \sum_{t'=t+1}^{T-2} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t'+1]} \frac{\partial \boldsymbol{S}^l[t'+1]}{\partial \boldsymbol{U}^l[t'+1]} \frac{\partial \boldsymbol{U}^l[t'+1]}{\partial \boldsymbol{m}^l[t']} \frac{\partial \boldsymbol{m}^l[t']}{\partial \mathbb{U}^l[t']} \prod_{t''=t+1}^{t'-1} \theta^l[t''] \\
&= \sum_{t'=t}^{T-2} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t'+1]} \frac{\partial \boldsymbol{S}^l[t'+1]}{\partial \boldsymbol{U}^l[t'+1]} \frac{\partial \boldsymbol{U}^l[t'+1]}{\partial \boldsymbol{m}^l[t']} \frac{\partial \boldsymbol{m}^l[t']}{\partial \mathbb{U}^l[t']} \prod_{t''=t+1}^{t'-1} \theta^l[t'']
\end{aligned} \tag{41}$$

By combining Equation (38) and Equation (41), we obtain the following expression for $t = T - 1, \ldots, 1$

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[T]} \frac{\partial \boldsymbol{S}^l[T]}{\partial \boldsymbol{U}^l[T]} \frac{\partial \boldsymbol{U}^l[T]}{\partial \boldsymbol{m}^l[T-1]} \frac{\partial \boldsymbol{m}^l[T-1]}{\partial \mathbb{U}^l[T-1]} + \sum_{t'=t}^{T-2} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t'+1]} \frac{\partial \boldsymbol{S}^l[t'+1]}{\partial \boldsymbol{U}^l[t'+1]} \frac{\partial \boldsymbol{U}^l[t'+1]}{\partial \boldsymbol{m}^l[t']} \frac{\partial \boldsymbol{m}^l[t']}{\partial \mathbb{U}^l[t']} \prod_{t''=t+1}^{t'-1} \theta^l[t''] \\
&= \sum_{t'=t}^{T-1} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t'+1]} \frac{\partial \boldsymbol{S}^l[t'+1]}{\partial \boldsymbol{U}^l[t'+1]} \frac{\partial \boldsymbol{U}^l[t'+1]}{\partial \boldsymbol{m}^l[t']} \frac{\partial \boldsymbol{m}^l[t']}{\partial \mathbb{U}^l[t']} \prod_{t''=t+1}^{t'-1} \theta^l[t'']
\end{aligned} \tag{42}$$

To further analyze the effect on the gradient, we examine each component of the equation.

$$\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} = \sum_{t'=t}^{T-1} \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t]} \underbrace{\frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]}}_{>0} \underbrace{\frac{\partial \boldsymbol{U}^l[t'+1]}{\partial \boldsymbol{m}^l[t']}}_{>0} \underbrace{\frac{\partial \boldsymbol{m}^l[t']}{\partial \mathbb{U}^l[t']}}_{<0} \prod_{t''=t+1}^{t'-1} \underbrace{\left(\underbrace{\frac{\partial \mathbb{U}^l[t''+1]}{\partial \mathbb{U}^l[t'']}}_{=1} + \underbrace{\xi^l[t''+1] \frac{\partial \boldsymbol{U}^l[t''+1]}{\partial \boldsymbol{m}^l[t'']} \frac{\partial \boldsymbol{m}^l[t'']}{\partial \mathbb{U}^l[t'']}}_{\in(-1,0)} \right)}_{>0} \tag{43}$$

According to the analysis in Lemma 1, in the output layer, $\frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^L[t]} = \boldsymbol{Y}^L[t] - \hat{\boldsymbol{Y}}^L[t]$, where $\boldsymbol{Y}^L[t]$ represents the average output (within [0,1]), and $\hat{\boldsymbol{Y}}^L[t]$ represents the predicted values encoded in a one-hot format (taking values of either 0 or 1). Therefore,

$$\frac{\partial \mathcal{L}}{\partial \mathbb{U}^L[t]} \begin{cases} \geq 0, & \hat{\boldsymbol{Y}}[t'] = 1 \\ \leq 0, & \hat{\boldsymbol{Y}}[t'] = 0 \end{cases} \tag{44}$$

Similarly, the gradient associated with the input current can be expanded as:

$$\frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]} \frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{I}^l[t+1]} \frac{\partial \boldsymbol{I}^l[t+1]}{\partial \mathbb{I}^l[t]} + \frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t+1]} \left(\frac{\partial \mathbb{I}^l[t+1]}{\partial \mathbb{I}^l[t]} + \frac{\partial \mathbb{I}^l[t+1]}{\partial \boldsymbol{I}^l[t+1]} \frac{\partial \boldsymbol{I}^l[t+1]}{\partial \mathbb{I}^l[t]} \right) \tag{45}$$

For the current additional derivative term, due to its rapid decay responsible for short-term adjustments and characterized by a very small coefficient, from $t = T-1, \ldots, 1$, the derivative can be approximated as:

$$\frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} \approx \frac{\partial \mathcal{L}}{\partial \boldsymbol{S}^l[t+1]} \underbrace{\frac{\partial \boldsymbol{S}^l[t+1]}{\partial \boldsymbol{U}^l[t+1]}}_{>0} \underbrace{\frac{\partial \boldsymbol{U}^l[t+1]}{\partial \boldsymbol{I}^l[t+1]}}_{>0} \underbrace{\frac{\partial \boldsymbol{I}^l[t+1]}{\partial \mathbb{I}^l[t]}}_{<0} \tag{46}$$

This follows the same sign pattern as the membrane potential derivative term:

$$\frac{\partial \mathcal{L}}{\partial \mathbb{I}^L[t]} \begin{cases} \geq 0, & \hat{\boldsymbol{Y}}[t'] = 1 \\ \leq 0, & \hat{\boldsymbol{Y}}[t'] = 0 \end{cases} \tag{47}$$

Similarly, based on Equation (28) and the chain rule, the sign of the additional gradient in the previous layers remains consistent with the weights, determined by the final output. Specifically, $\frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]}$ and $\frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]}$ are ≥ 0 if $\hat{\boldsymbol{Y}}[t'] = 1$, and ≤ 0 if $\hat{\boldsymbol{Y}}[t'] = 0$.

The gradient vanish problem is caused by the term $\epsilon^l[t]$, which tends toward zero during calculations according to Lemma 1. Here, we identify the additional gradient pathways in the expanded equation that facilitate gradient propagation during backpropagation. The additional gradient can be expressed as follows:

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} \xi^l[t] + \frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} \delta^l[t] &= \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} \left(\frac{\partial \mathbb{U}^l[t]}{\partial \bar{\boldsymbol{m}}^l[t]} \epsilon^l[t] + \frac{\partial \mathbb{U}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} \right) + \frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} \left(\frac{\partial \mathbb{I}^l[t]}{\partial \boldsymbol{S}^l[t]} \frac{\partial \boldsymbol{S}^l[t]}{\partial \boldsymbol{U}^l[t]} \right) \\ &= \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} \cdot \lambda_{\mathbb{U}} \cdot \boldsymbol{S}^l[t] \cdot \epsilon^l[t] + \frac{\partial \mathcal{L}}{\partial \mathbb{U}^l[t]} \cdot \lambda_{\mathbb{U}} \cdot \bar{\boldsymbol{m}}^l[t] + \frac{\partial \mathcal{L}}{\partial \mathbb{I}^l[t]} \cdot \lambda_{\mathbb{I}} \cdot \boldsymbol{I}^l[t-1] \quad \begin{cases} \geq 0, & \hat{\boldsymbol{Y}}[t'] = 1 \\ \leq 0, & \hat{\boldsymbol{Y}}[t'] = 0 \end{cases} \end{aligned} \tag{48}$$

E Pseudocode of ILIF mdoel

We provide the pseudocode of the ILIF model's spiking process in Algorithm 1. The red components represent the added inhibitory mechanisms compared to the vanilla LIF model. The abbreviations "P.L." and "P.T." denote the previous layer and previous time step, respectively.

Algorithm 1 Main Fire Procedure for LIF Model

```
Input: Total Time Steps $T$; Decay Rates $\lambda$, $\lambda_{\mathbb{U}}$, $\lambda_{\mathbb{I}}$; Threshold $V_{\text{th}}$
Initialize: $\boldsymbol{U}_0 \leftarrow 0$, $\mathbb{U}_0 \leftarrow 0$, $\boldsymbol{I}_0 \leftarrow 0$, $\mathbb{I}_0 \leftarrow 0$
 1: for $t = 1$ to $T$ do
 2:     $\boldsymbol{I} \leftarrow \boldsymbol{S}^{P.L.} \cdot W - \mathbb{I}^{P.T.}$          // Generate Input Current with Feedback Inhibition
 3:     $\boldsymbol{U} \leftarrow \lambda \cdot \boldsymbol{m}^{P.T.} + \boldsymbol{I}$          // Update Membrane Potential
 4:     $\boldsymbol{S} \leftarrow \mathbb{H}(\boldsymbol{U} - V_{\text{th}})$          // Check Threshold and Fire
 5:     $\bar{\boldsymbol{m}} \leftarrow \boldsymbol{U} - \boldsymbol{S} \cdot V_{\text{th}}$          // Reset
 6:     $\mathbb{U} \leftarrow \lambda_{\mathbb{U}} \cdot \mathbb{U}^{P.T.} + \boldsymbol{S} \cdot \bar{\boldsymbol{m}}$          // Update Membrane Potential Inhibition Unit
 7:     $\boldsymbol{m} \leftarrow \bar{\boldsymbol{m}} - \boldsymbol{S} \cdot \sigma(\mathbb{U})$          // Inhibitory Reset
 8:     $\mathbb{I} \leftarrow \lambda_{\mathbb{I}} \cdot \mathbb{I}^{P.T.} + \boldsymbol{S} \cdot \boldsymbol{I}$          // Update Current Inhibition Unit
 9:     Output: $\boldsymbol{S}$
10: end for
```

F Detailed Experimental Settings and Dateset Description

We utilize rectangle surrogate functions with $\lambda = 1$ and $\tau = 1.1$, ensuring consistent random seed values of 1234. No data augmentation is applied across any of the datasets. All experiments are conducted on an NVIDIA RTX 4090D GPU. Detailed descriptions of the dataset configurations and default experimental setups are provided as shown in Table 4.

Table 4: Hyperparameter settings for various datasets.

Dataset	BS	Epochs	LR	Optimizer	Weight Decay	Dropout	$\mathbb{U}$	$\mathbb{I}$
CIFAR10	128	200	0.1	SGD	5e-05	0.1	1	0.03
CIFAR100	128	200	0.1	SGD	5e-04	0.1	1	0.03
DVSCIFAR10	128	200	0.05	SGD	5e-04	0.3	1	0.03
DVSGesture	16	200	0.1	SGD	5e-04	0.4	1	0.05

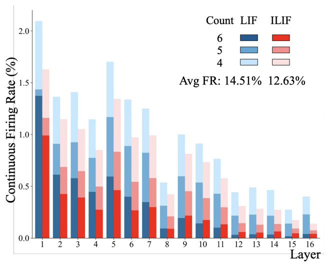
CIFAR10 and CIFAR100 are benchmark datasets for image classification. CIFAR10 consists of 10 classes with 60,000 32 $\times$ 32 color images, divided into 50,000 training and 10,000 testing samples. CIFAR100 extends this to 100 classes grouped into 20 superclasses, with the same total number of images but fewer per class, making it more challenging. Direct encoding is utilized to convert image pixels into time series, with pixel values repeatedly fed into the input layer at each timestep.

DVSCIFAR10 is an event-based dataset derived from CIFAR10 using a Dynamic Vision Sensor (DVS). It represents images as asynchronous event streams with spatial (x, y), polarity, and timestamp information. The dataset comprises 10,000 event-based images, where the pixel resolution has been expanded to 128×128.

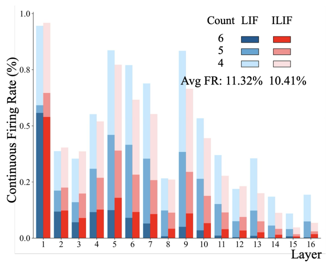
DVSGesture is also an event-based dataset comprising 11 hand gesture categories, performed by 29 participants under three different lighting conditions. It includes a total of 1,464 samples, with 1,176 samples in the training set and 288 samples in the test set. Each sample has a fixed resolution of 128x128 pixels.

G Continuous Firing Rate

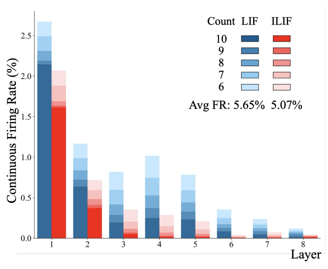
This is a comparison of the continuous firing rates across different datasets and layers between the LIF and ILIF models. The ILIF model consistently exhibits lower continuous firing rates across almost all layers compared to the LIF model, demonstrating its inhibitory effect. Simultaneously, the ILIF model demonstrates lower average firing rates than the LIF model, with DVSGesture showing a significant reduction of over 30%, as illustrated in Figure 6.



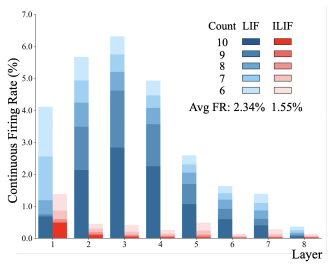
(a) CIFAR10



(b) CIFAR100



(c) DVSCIFAR10



(d) DVSGesture

Figure 6: Continuous firing rate comparison across different datasets

H Energy Consumption

To assess the energy efficiency of SNN, we adopt a standard approach from neuromorphic computing that estimates total synaptic operation power (SOP) based on the number of fundamental operations and their associated energy costs [Zhou *et al.*, 2022]. Specifically, SOP is calculated as:

$$\text{SOP}_s = E_{\text{AC}} \cdot AC_s + E_{\text{MAC}} \cdot MAC_s \tag{49}$$

where E_{AC} and E_{MAC} denote the energy consumption per accumulation (AC) and multiply-accumulate (MAC) operation, respectively. Following the energy model in [Han *et al.*, 2015], we assume each 32-bit floating-point addition consumes 0.9 picojoules (pJ), and each MAC operation consumes 4.6 pJ.

In SNN, neurons transmit binary spike signals, $s_i^l[t] \in \{0, 1\}$, indicating whether neuron i in layer l fires at time step t. A spike activates all its outgoing synapses, each performing an addition. If a neuron has f_i^l outgoing connections (fan-out), the total number of AC operations across the network is given by:

$$AC_s = \sum_{t=1}^{T} \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l \cdot s_i^l[t] \tag{50}$$

Here, T is the total number of time steps, L is the number of layers, and N^l is the number of neurons in layer l.

By contrast, ANN compute statically without temporal dynamics. Each neuron performs a single forward pass involving a fixed number of MAC operations, determined solely by its synaptic connections:

$$MAC_s = \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l \tag{51}$$

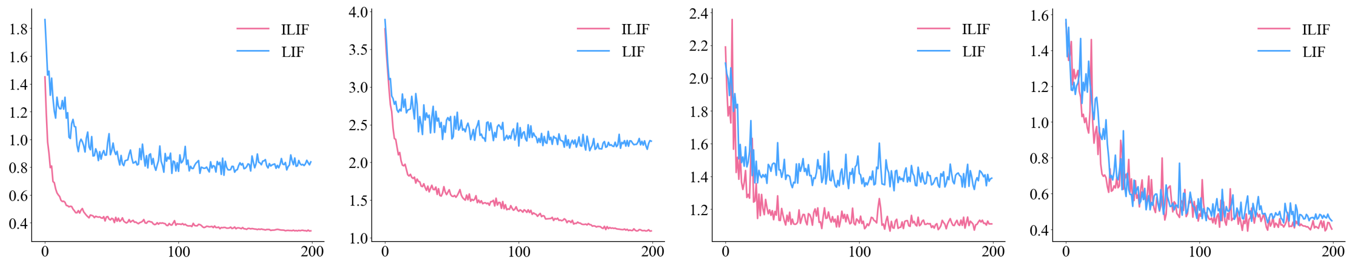
Combining these operation counts with their energy costs yields the total SOP for any given network configuration. As shown in Table 5, SNNs demonstrate a clear advantage in energy efficiency over ANNs. Compared to LIF, ILIF incurs more MAC operations due to the inclusion of the inhibitory unit. However, its significantly lower spike rate not only offsets this overhead but leads to an overall reduction in energy consumption. On the DVS-Gesture dataset, ILIF shows slightly higher energy usage than LIF, attributed to the dataset's higher resolution and extended temporal length. Despite this, ILIF consistently achieves better representational accuracy with fewer spikes and lower energy cost overall. These results highlight the effectiveness of incorporating inhibitory mechanism in achieving energy-efficient yet expressive spiking models.

Table 5: Comparison of Synaptic Operations and Energy Consumption Across ANN, LIF, and ILIF

Dataset	Network Architecture	Method	T	ACs (M)	MACs (M)	SOP Energy (μJ)
CIFAR10	ResNet18	ANN	1	0	549.13	2525.980
		LIF	6	568.26	3.34	526.806
		ILIF	6	488.12	6.68	470.056
CIFAR100	ResNet18	ANN	1	0	549.13	2525.980
		LIF	6	438.70	3.34	410.203
		ILIF	6	409.11	6.68	398.951
DVSCIFAR10	VGG11	LIF	10	201.58	3.41	197.106
		ILIF	10	157.40	6.82	173.031
DVSGesture	VGG11	LIF	20	820.05	48.50	961.133
		ILIF	20	653.86	96.99	1034.647

I Loss Curve

As shown in Figure 7, the loss curves of the vanilla LIF and ILIF models indicate that the vanilla LIF model undergoes fluctuations during training—mainly due to excessive neuron activation and gradient loss across time—thereby converging more slowly and ending at a higher final loss. By contrast, the ILIF model exhibits a smoother, steadily declining loss curve, as its inhibitory units provide shortcuts for rapid gradient backpropagation and help suppress activated neurons. This design ultimately enables the ILIF model to converge to a much lower loss value and attain higher accuracy.



(a) CIFAR10 (b) CIFAR100 (c) DVSCIFAR10 (d) DVSGesture

Figure 7: Loss curve comparison across different datasets

J Comparison with SOTA

Due to space limitations, additional experiments with mainstream architectures are presented in this section, demonstrating the excellent performance of our method across various network architectures, as shown in Table 6. Models with higher accuracy achieved through data augmentation techniques are excluded from comparison. Notably, our method surpasses the traditional accuracy benchmarks on DVSCIFAR10 and DVSGesture datasets, achieving superior performance with only 200 iterations (compared to the conventional 300 iterations).

Table 6: Accuracy Comparison on CIFAR10, CIFAR100, DVSCIFAR10 and DVSGesture

Dataset	Method	Network Architecture	Time Step	Accuracy (%)
CIFAR10	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	4 / 6	92.92 / 93.16
	SEW-ResNet [Fang *et al.*, 2021a]	SEW-ResNet-18	4	91.22 / 94.39
	Dspike [Li *et al.*, 2021]	Modified ResNet-18	4 / 6	93.66 / 94.05
	TET [Deng *et al.*, 2022b]	ResNet-19	4 / 6	94.44 / 94.50
	SLTT* [Meng *et al.*, 2023]	ResNet-18	6	94.44
	GLIF [Yao *et al.*, 2022]	ResNet-18	4	94.67
	SML [Deng *et al.*, 2023]	ResNet-18	6	95.12
	CLIF [Huang *et al.*, 2024]	ResNet-18	4 / 6	94.89 / 95.41
	Ours	ResNet-18	4 / 6	**95.24 / 95.49**
	Temporal Pruning [Chowdhury *et al.*, 2022]	VGG-16	5	93.44
	SLTT* [Meng *et al.*, 2023]		6	93.28
	Ours		6	**94.25**
CIFAR100	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-19	4 / 6	70.86 / 71.12
	TET [Deng *et al.*, 2022b]	ResNet-19	4 / 6	74.74 / 74.72
	Dspike [Li *et al.*, 2021]	Modified ResNet-18	4 / 6	73.35 / 74.24
	SLTT* [Meng *et al.*, 2023]	ResNet-18	6	74.38
	GLIF [Yao *et al.*, 2022]	ResNet-18	4 / 6	76.42 / 77.28
	Ours	ResNet-18	4 / 6	**77.43 / 78.51**
	Temporal Pruning [Chowdhury *et al.*, 2022]	VGG-16	5	71.58
	SLTT* [Meng *et al.*, 2023]		6	72.55
	Ours		6	**75.25**
DVSCIFAR10	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-18	10	67.80
	Dspike [Li *et al.*, 2021]		10	75.40
	InfLoR [Guo *et al.*, 2022]		10	75.50
	Ours		10	**77.71**
	OTTT* [Xiao *et al.*, 2022]	VGG-11	10	76.27
	DSR [Meng *et al.*, 2022]		20	77.27
	SLTT [Meng *et al.*, 2023]		10	77.17
	TET [Deng *et al.*, 2022b]		10	77.33
	Ours		10	**78.60**
DVSGesture	STBP-tdBN [Zheng *et al.*, 2021]	ResNet-18	40	96.87
	Ours		20	**96.88**
	OTTT [Xiao *et al.*, 2022]	VGG-11	20	96.88
	SLTT [Meng *et al.*, 2023]		20	97.92
	CLIF [Huang *et al.*, 2024]		20	97.92
	Ours		20	**97.92**

* indicates that data augmentation was used.

Ethical Statement

There are no ethical issues.

Acknowledgments

The work was supported in part by start-up funds with No. MSRI8001004 and No. MSRI9002005.

References

[Bellec *et al.*, 2018] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. *NeurIPS*, 31, 2018.

[Cai and Li, 2021] Zongyuan Cai and Xinze Li. Neuromorphic brain-inspired computing with hybrid neural networks. In *2021 IEEE International Conference on Artificial Intelligence and Industrial Design (AIID)*, pages 343–347. IEEE, 2021.

[Chowdhury *et al.*, 2022] Sayeed Shafayet Chowdhury, Nitin Rathi, and Kaushik Roy. Towards ultra low latency spiking neural networks for vision and sequential tasks using temporal pruning. In *ECCV*, pages 709–726. Springer, 2022.

[Dampfhoffer *et al.*, 2022] Manon Dampfhoffer, Thomas Mesquida, Alexandre Valentian, and Lorena Anghel. Investigating current-based and gating approaches for accurate and energy-efficient spiking recurrent neural networks. In *International Conference on Artificial Neural Networks*, pages 359–370. Springer, 2022.

[Deng *et al.*, 2022a] Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. *ICLR*, 2022.

[Deng *et al.*, 2022b] Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting. *ICLR*, 2022.

[Deng *et al.*, 2023] Shikuang Deng, Hao Lin, Yuhang Li, and Shi Gu. Surrogate module learning: Reduce the gradient error accumulation in training spiking neural networks. In *ICML*, pages 7645–7657. PMLR, 2023.

[Ding *et al.*, 2022] Jianchuan Ding, Bo Dong, Felix Heide, Yufei Ding, Yunduo Zhou, Baocai Yin, and Xin Yang. Biologically inspired dynamic thresholds for spiking neural networks. *NeurIPS*, 35:6090–6103, 2022.

[Duan *et al.*, 2022] Chaoteng Duan, Jianhao Ding, Shiyan Chen, Zhaofei Yu, and Tiejun Huang. Temporal effective batch normalization in spiking neural networks. *NeurIPS*, 35:34377–34390, 2022.

[Fang *et al.*, 2020] Haowen Fang, Amar Shrestha, Ziyi Zhao, and Qinru Qiu. Exploiting neuron and synapse filter dynamics in spatial temporal learning of deep spiking neural network. *arXiv preprint arXiv:2003.02944*, 2020.

[Fang *et al.*, 2021a] Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *NeurIPS*, 34:21056–21069, 2021.

[Fang *et al.*, 2021b] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothée Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. *ICCV*, pages 2661–2671, 2021.

[Guo *et al.*, 2022] Yufei Guo, Yuanpei Chen, Liwen Zhang, YingLei Wang, Xiaode Liu, Xinyi Tong, Yuanyuan Ou, Xuhui Huang, and Zhe Ma. Reducing information loss for spiking neural networks. In *ECCV*, pages 36–52. Springer, 2022.

[Guo *et al.*, 2023] Yufei Guo, Yuhan Zhang, Yuanpei Chen, Weihang Peng, Xiaode Liu, Liwen Zhang, Xuhui Huang, and Zhe Ma. Membrane potential batch normalization for spiking neural networks. In *ICCV*, pages 19420–19430, 2023.

[Han *et al.*, 2015] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28, 2015.

[Huang *et al.*, 2024] Yulong Huang, Xiaopeng Lin, Hongwei Ren, Haotian Fu, Yue Zhou, Zunchang Liu, Biao Pan, and Bojun Cheng. Clif: Complementary leaky integrate-and-fire neuron for spiking neural networks. *ICML*, 2024.

[Jiang *et al.*, 2024] Haiyan Jiang, Vincent Zoonekynd, Giulia De Masi, Bin Gu, and Huan Xiong. Tab: Temporal accumulated batch normalization in spiking neural networks. In *ICLR*, 2024.

[Li *et al.*, 2021] Yuhang Li, Yufei Guo, Shanghang Zhang, Shikuang Deng, Yongqing Hai, and Shi Gu. Differentiable spike: Rethinking gradient-descent for training spiking neural networks. *NeurIPS*, 34:23426–23439, 2021.

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

[Meng *et al.*, 2022] Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Training high-performance low-latency spiking neural networks by differentiation on spike representation. In *CVPR*, pages 12444–12453, 2022.

[Meng *et al.*, 2023] Qingyan Meng, Mingqing Xiao, Shen Yan, Yisen Wang, Zhouchen Lin, and Zhi-Quan Luo. Towards memory-and time-efficient backpropagation for training spiking neural networks. In *ICCV*, pages 6166–6176, 2023.

[Neftci *et al.*, 2019] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

[Niu and Wei, 2023] Li-Ye Niu and Ying Wei. Cirm-snn: Certainty interval reset mechanism spiking neuron for enabling high accuracy spiking neural network. *Neural Processing Letters*, 55(6):7561–7582, 2023.

[Tavanaei *et al.*, 2019] Amirhossein Tavanaei, Masoud Ghodrati, Saeed Reza Kheradpisheh, Timothée Masquelier, and Anthony Maida. Deep learning in spiking neural networks. *Neural networks*, 111:47–63, 2019.

[Wang and Yu, 2024] Lihao Wang and Zhaofei Yu. Autaptic synaptic circuit enhances spatio-temporal predictive learning of spiking neural networks. *PMLR*, 2024.

[Wei *et al.*, 2023] 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 *ICCV*, pages 10552–10562, 2023.

[Werbos, 1990] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[Wu *et al.*, 2019] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *AAAI*, volume 33, pages 1311–1318, 2019.

[Xiao *et al.*, 2022] Mingqing Xiao, Qingyan Meng, Zongpeng Zhang, Di He, and Zhouchen Lin. Online training through time for spiking neural networks. *NeurIPS*, 35:20717–20730, 2022.

[Yao *et al.*, 2022] Xingting Yao, Fanrong Li, Zitao Mo, and Jian Cheng. Glif: A unified gated leaky integrate-and-fire neuron for spiking neural networks. *NeurIPS*, 35:32160–32171, 2022.

[Zenke and Ganguli, 2018] Friedemann Zenke and Surya Ganguli. Superspike: Supervised learning in multilayer spiking neural networks. *Neural computation*, 30(6):1514–1541, 2018.

[Zhang *et al.*, 2019] Anguo Zhang, Hongjun Zhou, Xiumin Li, and Wei Zhu. Fast and robust learning in spiking feed-forward neural networks based on intrinsic plasticity mechanism. *Neurocomputing*, 365:102–112, 2019.

[Zheng *et al.*, 2021] Hanle Zheng, Yujie Wu, Lei Deng, Yifan Hu, and Guoqi Li. Going deeper with directly-trained larger spiking neural networks. In *AAAI*, volume 35, pages 11062–11070, 2021.

[Zhou *et al.*, 2022] Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022.