LASER: A HIGH-FIDELITY SPIKE REPRESENTATION SNN FRAMEWORK WITH SURROGATE-FREE TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking Neural Networks (SNNs), as the third generation of neural networks inspired by biological neural systems, demonstrate great potential for energy-efficient computing due to their inherent event-driven sparsity. However, a long-standing core challenge of SNNs lies in the intrinsic error introduced when approximating continuous values with discrete spikes, which makes it difficult to match the accuracy of Artificial Neural Networks (ANNs). To address this issue, we propose a high-fidelity spike representation SNN framework with surrogate-free training, called LASER. Specifically, LASER introduces a precise bi-directional mapping scheme between discrete and continuous values for linear computation. In addition, we design a piecewise-approximate spiking representation for nonlinear functions, enabling high-fidelity forward propagation. Building on this, we propose an STE-based backpropagation strategy to ensure functional consistency with ANNs and to achieve stable training for SNNs. Experiments validate the overall framework, showing that, unlike prior methods where errors diffuse across the network, LASER confines a slight error (0.6%) solely to the non-linear module. LASER achieves over 50% lower perplexity compared to the state-of-the-art SNN framework, with almost lossless accuracy.

1 INTRODUCTION

As the third generation of neural network models, Spiking Neural Networks (SNNs) distinguish themselves from Artificial Neural Networks (ANNs) by processing information via discrete spikes rather than continuous values, thereby closely emulating the firing dynamics of biological neurons. This event-driven mechanism grants SNNs a critical advantage in energy efficiency, positioning them as a compelling alternative for low-power applications and dedicated neuromorphic hardware (Maass, 1997; Roy et al., 2019; Davies et al., 2018; 2021).

However, the binary nature of spikes constrains the information representation capacity of SNN (Auge et al., 2021; Guo et al., 2021). This limitation compromises the achievement of both high-fidelity spike representation and the functionality required to match ANNs. ① During the encoding stage, converting continuous values into discrete spikes inevitably introduces approximation errors. Even worse, not all nonlinear function can be effectively represented by spikes (e.g., Softmax). ②In addition, enabling backpropagation for SNNs is essential for training but is complicated by the non-differentiable nature of spikes.

To achieve the high-fidelity spike representation and functionality, existing works typically rely on computation-intensive and surrogate solutions, which come at the cost of either efficiency or accuracy. First, existing works rely on additional computation to compensate for errors introduced during the encoding process. Specifically, longer timesteps are often employed, which significantly increases computational latency. As shown in Table 1, in classical TTFS encoding (Bonilla et al., 2022; Stanojevic et al., 2023), increasing the timestep from 16 to 1,024 reduces the error while increasing computation latency 64-fold. Second, these works use surrogate techniques to enable spike representation of nonlinear functions. For example, they replace activation functions with "spike-friendly" ReLU (Zhou et al., 2022; Yao et al., 2023; Hwang et al., 2024). This strategy alters the network architecture, undermining the carefully optimized performance achieved on the

Table 1: Reconstruction fidelity (MSE; mean $\pm$ std over 10,000 random values) with explicit precision and latency cost (Time Steps). BSE attains 10^{11}–10^{18} lower error than rate coding and TTFS under the same step budgets, reaching near-machine precision at FP32. Notably, TTFS requires as many as 1,024 time steps to match the fidelity that BSE already achieves within 16 steps, highlighting the substantially higher latency cost of TTFS.

Encoding Scheme	Time Steps	MSE (mean $\pm$ std)
Rate Coding	16	$7.70 \pm 0.02 \times 10^{4}$
Rate Coding	32	$4.46 \pm 0.01 \times 10^{4}$
TTFS	16	$3.68 \pm 0.02 \times 10^{5}$
TTFS	32	$6.03 \pm 0.01 \times 10^{4}$
TTFS	1024	$1.03 \pm 0.01 \times 10^{-9}$
Ours	2	$\mathbf{9.26 \pm 0.04 \times 10^{4}}$
Ours	4	$\mathbf{3.70 \pm 0.03 \times 10^{-4}}$
Ours	8	$\mathbf{1.28 \pm 0.01 \times 10^{-6}}$
Ours	16	$\mathbf{1.03 \pm 0.02 \times 10^{-9}}$
Ours	32	$\mathbf{1.33 \pm 0.01 \times 10^{-14}}$

original network design. Finally, these works using surrogate gradients or soft substitutes to achieve backpropagation based on spike representation (Neftci et al., 2019; Shrestha & Orchard, 2018; Deng et al., 2022). Such approximation methods introduce extra errors, making training unstable.

In this paper, we propose LASER, a novel framework that achieves high-fidelity spike representation, maintains structural consistency with ANNs, and enables surrogate-free training. LASER proposes a new spike expression language that establishes a precise bidirectional mapping between discrete spike sequences and continuous values (i.e., encoding). Based on this language, it breaks away from the conventional paradigm of using surrogates to support both nonlinear computation and backpropagation. The main contributions are as follows:

- We first propose Bit Spike Encoding (BSE), an exact correspondence scheme between discrete spikes and continuous values. Its core idea is to transform floating-point numbers into integers through same-bitwidth quantization, preserving the same information entropy and machine precision as the original values (proof in Appendix A), and then map each integer to a spike sequence consistent with its binary representation within a fixed short time window.
- Building on BSE, we introduce the Adaptive Spiking Neural Codec (ASNC) for nonlinear computations. Unlike existing methods that require large-scale structural replacements, ASNC functions as a structural approximator that locally approximates arbitrary nonlinear functions in the spike domain without modifying the network architecture, keeping the overall structural path consistent and avoiding the need for training alignment.
- Building on the high-fidelity forward propagation, we introduce the Straight-Through Estimator (STE) to replace derivatives. In this case, the identity gradient surrogate is no longer an empirical or heuristic approximation, but an equivalent approximation strictly consistent with the forward numerical path. Since the gradient path is fully aligned with the numerical path, backpropagation does not introduce additional modifications to the computation process, thereby ensuring stable convergence in training and preventing error accumulation and diffusion.

Our experiments systematically validate the effectiveness of our method. As a result, in large-model comparisons, on LLaMA-2 7B the perplexity increase is 0.05% of that of the baseline method; at the 70B scale in end-to-end experiments, the overall accuracy drop is below 2%. On neuromorphic hardware, energy consumption on Loihi 2 decreases 98% comparing to that on the GPU.

2 Related Works

Linear encoding. Existing spike encoding methods essentially use spike sequences to approximate continuous values for information representation, but they suffer from large errors and high

latency. For example, using rate coding (Rueckauer et al., 2017), 1,000 timesteps are required to approximate 0.625, which leads to low information density and poor expressive efficiency, amplifies jitter, and slows convergence (Auge et al., 2021; Guo et al., 2021), making it difficult to apply in large-scale modeling scenarios. Recent studies attempt to trade efficiency for accuracy. TTFS-Former encoding achieves accuracy close to ANNs (Zhao et al., 2025), but due to the single-spike constraint, it still requires 1,024 steps to reach FP16 precision (Stanojevic et al., 2023). Methods such as Spikformer/Spikformer-v2 (Zhou et al., 2022; 2024) and SpikeZIP/SpikeZIP-TF (You et al., 2024) also rely on long time windows or high firing rates to maintain accuracy. Rather than reducing the losses caused by such approximations, we propose Bit Spike Encoding (BSE), a bidirectional exact mapping between spike sequences and continuous values, which enables spike sequences to precisely represent continuous values within a fixed short time window, thereby eliminating the error propagation path in network linear computations.

Nonlinear computation. Because existing methods in handling nonlinear computations either design dedicated approximation modules for certain linear operations, or convert some modules into "spike-friendly" substitute structures with additional training alignment, they inevitably introduce errors that affect subsequent computations. Spiking Self-Attention (SSA) (Zhou et al., 2022) bypasses non-spike operators through structural rewriting (Yao et al., 2023; Hwang et al., 2024; Wang et al., 2023). Some works also directly provide spike-domain equivalent forms of Softmax and LayerNorm (You et al., 2024; Tang et al., 2024) to support Transformer conversion, while SpikeGPT (Zhu et al., 2024) and Spiking-LLM (Xing et al., 2025) adopt soft gating and approximately differentiable units for modeling. Rather than designing specialized approximations or substitutes, we propose a general structural approximator, Adaptive Spiking Neural Codec (ASNC), which performs piecewise approximation for arbitrary nonlinear computations while keeping inputs and outputs in the BSE format, so that the potential path of error accumulation and diffusion is cut off by subsequent linear computations, forming a localized error mechanism.

Training. The gradient computation of SNNs is limited by the non-differentiability of spikes, and replacing non-differentiable functions with surrogate gradients introduces additional errors that affect training convergence. SpikeLM (Xing et al., 2024) attempts direct training, relying on surrogates or soft substitutes to ensure differentiability, but inevitably introduces errors. STBP (Wu et al., 2018), SLAYER (Shrestha & Orchard, 2018), DIET-SNN (Rathi & Roy, 2020), and conversion with fine-tuning (Rathi et al., 2020) often require multi-stage procedures or special structures, and introduce surrogate gradients at multiple levels, reducing training stability. Our approach instead introduces a lightweight Straight-Through Estimator (STE) on top of the high-fidelity and structurally consistent forward path guaranteed by BSE and ASNC. Although used globally, since errors are strictly confined to the single nonlinear module and the gradient path is fully aligned with the numerical path, backpropagation does not introduce additional modifications to the computation process, thereby ensuring stable training convergence.

3 Methodology: A Framework for Precise Spiking Computation

We aim to answer a core question: whether it is possible to construct a high-fidelity structural mapping mechanism from ANNs to SNNs, so that as the model scales up, it can still maintain accuracy consistency, error controllability, and training stability. We propose a three-step roadmap, aiming to construct an ideal SNN through high-fidelity mapping, structural consistency, and error controllability. To achieve this goal, we propose Bit Spike Encoding (BSE), a bidirectional exact mapping between spike sequences and continuous values without sacrificing efficiency, to maximally preserve information expression; a general structural approximator, Adaptive Spiking Neural Codec (ASNC), to realize nonlinear modeling, which requires no structural modifications and localizes errors; and, under high-fidelity and structurally consistent forward propagation, to introduce an identity-gradient proxy (STE). Since the gradient path is fully aligned with the numerical path, backpropagation does not introduce additional modifications to the computation process, thereby ensuring stable convergence of training.

3.1 Bit Spike Encoding (BSE)

Existing encoding schemes generally compress the continuous space into a limited representation, leading to low precision and significant information loss, thereby forcing a trade-off of efficiency

for accuracy. Instead of data compression, we propose a novel spike representation language that goes beyond simple compression—a bidirectional precise mapping between spike sequences and continuous values, termed **Bit Spike Encoding (BSE)**. BSE enables spike sequences to accurately represent continuous values within a fixed low-latency time window, thereby eliminating error propagation paths in the linear computations of spiking neural networks.

BSE relies on an Integrate-and-Fire (IF) neuron with soft reset and a dynamic threshold. Consider a single scalar element. The forward pass is given in Equation 1:

$$\begin{aligned} V_m(t) &= \sum_{\tau<t} I_{\text{in}}(\tau) \;-\; \sum_{\tau<t} S_{\text{actual}}(\tau)\,\Theta(\tau), \\ I_{\text{out}}(t) &= \begin{cases} 1, & \text{if } V_m(t) + I_{\text{in}}(t) \geq \Theta(t), \\ 0, & \text{otherwise,} \end{cases} \\ S_{\text{actual}}(t) &= I_{\text{out}}(t), \\ V_m(t{+}1) &= V_m(t) + I_{\text{in}}(t) - I_{\text{out}}(t)\,\Theta(t). \end{aligned} \tag{1}$$

where $I_{\text{in}}(t)$ is the input current, $V_m(t)$ is the membrane potential, $S_{\text{actual}}(t)$ is the emitted spike, $\Theta(t)$ is the dynamic threshold, $I_{\text{out}}(t)$ is the thresholding indicator, and t, τ are discrete time indices. Then we define the Soma compartment, whose body consists of a current accumulation cavity ($\mathcal{A}$) and a synaptic strength in the time domain.

For Soma, the time-domain synaptic strength at step t is defined as:

$$\mathcal{G}(t) = W \cdot W_{\text{bit}}(t), t = 0, \ldots, N-1. \tag{2}$$

Here, W is the scalar weight, $W_{\text{bit}}(t)$ is the bit weight at step t, and $\mathcal{G}(t)$ is the synaptic strength at step t. The input to Soma is given by the upstream spike $S_a(t) \in \{0, 1\}$ at step t, the offset $\mu \in \mathbb{R}$, the scaling factor $\lambda \in \mathbb{R}_{>0}$, and the bias $\beta \in \mathbb{R}$. Its forward pass is given in Equation 3:

$$\begin{aligned} Q_a &= \sum_{t=0}^{N-1} S_a(t)\, W_{\text{bit}}(t), \\ A &= Q_a \cdot W, \\ Y &= \frac{Q_a}{\lambda} W \;-\; W\mu \;+\; \beta, \\ I_{\text{out}} &= \min \mathcal{S}, \qquad \mu_{\text{out}} = -I_{\text{out}}, \\ R_{\text{out}} &= \max\{\, y + \mu_{\text{out}} : \, y \in \mathcal{S} \,\} + \varepsilon, \qquad \lambda_{\text{out}} = \frac{2^N - 1}{R_{\text{out}}}, \\ Q_y &= \text{clip}_{[0,\, 2^N - 1]}(\text{round}(\lambda_{\text{out}}\,(Y + \mu_{\text{out}}))) \,. \end{aligned} \tag{3}$$

Here, $S_a(t) \in \{0, 1\}$ is the input spike at step t, Q_a is the integer code in the N-step window, A is the accumulated current, Y is the de-scaled and de-biased real value, λ and μ are the input scale and offset, W is the weight, β is the bias, $\mathcal{S}$ is the set of observed Y values used for normalization, I_{out} is the minimum of $\mathcal{S}$, μ_{out} is its negation, R_{out} is the shifted range with small $\varepsilon > 0$, λ_{out} is the resulting scale, Q_y is the quantized integer current, and clip and round are elementwise clipping and rounding.

The single layer BSE encoding is defined in Equation 4:

$$\begin{aligned} (Q_y, \lambda_{\text{out}}, \mu_{\text{out}}) &= \text{Soma}\big(\{S_a(t)\}_{t=0}^{N-1};\, W,\, \lambda,\, \mu,\, W_{\text{bit}},\, \beta\big), \\ \Theta(t) &= W_{\text{bit}}(t), \qquad V_m(0) = 0, \qquad I_{\text{in}}(t) = \delta_{t0}\, Q_y, \\ I_{\text{out}}(t) &= \begin{cases} 1, & \text{if } V_m(t) + I_{\text{in}}(t) \geq \Theta(t), \\ 0, & \text{otherwise,} \end{cases} \\ S_y(t) &= I_{\text{out}}(t), \\ V_m(t{+}1) &= V_m(t) + I_{\text{in}}(t) - I_{\text{out}}(t)\,\Theta(t). \end{aligned} \tag{4}$$

Here, $\text{Soma}(\cdot)$ is the mapping defined above. Q_y is the integer current, λ_{out} and μ_{out} are the per-output scale and offset, $W_{\text{bit}}(t)$ is the bit weight, $\Theta(t)$ is the threshold, $V_m(t)$ is the membrane

potential, $I_{\text{in}}(t)$ is the input current with δ_{t0} being the Kronecker delta, $I_{\text{out}}(t)$ is the thresholding indicator, and $S_y(t)$ is the emitted spike sequence over the N-step window.

For other computations operating on the network's activation hierarchy—activation–activation multiplications such as the *Query–Key (QK)* matrix multiplication in self-attention—we adopt a "decode–compute–re-encode" strategy. This is a deliberate design choice, not a compromise. First, the two incoming BSE spike trains are losslessly decoded back to their corresponding numerical values. Then, the multiplication is performed using standard ANN multiply–accumulate (MAC) operations. Finally, in a crucial last step, the resulting numerical value is immediately re-encoded by a BSE encoder into a new, high-fidelity spike train. This strategy guarantees that such activation-level computations are mathematically equivalent to those in a quantized ANN, thus eliminating by design one of the primary sources of error in traditional A2S methods. By contrast, weight–activation multiplications (e.g., linear projections) can be directly supported in the BSE domain without this mechanism.

Experiments 4.2 show that when the BSE time window length equals the bit width of the original floating-point or fixed-point representation, for example FP16 with 16 bits, BSE encoding does not introduce errors that diffuses through the network. We also provide a mathematical analysis in the appendix proving that under this setting BSE is entropy-preserving. Hence ANN activations can be rendered as sparse and ordered spike trains in the time domain within a fixed window, achieving a reconstruction of information density from the numeric domain to the spiking domain. See Appendix A for the full proof.

In summary, BSE achieves deterministic and reversible encoding within a fixed time window, ensuring that linear computations remain strictly equivalent without introducing diffusive errors. This mechanism not only reconstructs information density in the spiking domain but also establishes a unified and stable representational language for the entire system. Therefore, BSE is not only the prerequisite for localized error confinement in ASNC but also the fundamental component that guarantees global structural consistency and training controllability.

3.2 Adaptive Spiking Neural Codec (ASNC)

In existing methods for handling nonlinear computations, the mainstream approach is structural substitution, which inevitably introduces additional errors; some works, although avoiding modification of the overall structure, design dedicated approximation modules for certain specific nonlinearities, but these lack generality. To address this, we propose a universal nonlinear approximation module that requires no structural replacement—the Adaptive Spiking Neural Codec (ASNC). ASNC can perform piecewise fitting for arbitrary nonlinear computations to preserve structural consistency. For each activated segment, we train it to converge to the corresponding nonlinear function piece, and immediately re-encode the computation result into BSE format, ensuring that all intermediate tensors remain in the spiking domain. Since the input distribution of nonlinear computations in neural networks follows a bounded Gaussian distribution (He et al., 2024; Lin et al., 2024), we further partition the neuron response intervals according to the input distribution: sensitive regions (including high-density areas and outlier intervals) are partitioned more densely for fine-grained fitting, while low-density, less-sensitive regions are partitioned more sparsely for coarse-grained fitting. This approach significantly reduces latency and overall loss while maintaining accuracy. See Appendix B for the rigorous proof that ASNC achieves a tighter error bound under Gaussian inputs than existing methods (such as SpikeZIP-FT (You et al., 2024)).

The codec mapping for segment i is defined as shown in Equation 5:

$$\text{spikes}_i(t) = \text{LIF}\big(\text{quantize}(x\, s_i + b_i),\, \theta_i(t)\big), \qquad \mathcal{C}_i(x) = \sum_{t=1}^{T} w_i(t)\, \text{spikes}_i(t), \tag{5}$$

where x is the input, s_i, b_i are affine parameters, $\theta_i(t)$ is the firing threshold, $w_i(t)$ are trainable decoding weights, t indexes the time step, T is the fixed time window, and $\mathcal{C}_i(x)$ is the decoded analog estimate.

Segment outputs are preserved in BSE form via direct re-encoding, as shown in Equation 6:

$$S^{\text{BSE}}(t) = \sum_{i=1}^{N} \mathcal{I}_i(x)\, S_i^{\text{BSE}}(t), \tag{6}$$

where $\mathcal{I}_i(x)$ selects the active segment(s), $S_i^{\mathrm{BSE}}(t)$ denotes the BSE spike train for segment i, and N is the total number of segments. This alignment ensures that the approximation error remains confined to the nonlinear unit.

To preserve fidelity with minimal complexity, segments are adaptively split when their normalized difficulty surpasses a threshold, as shown in Equation 7:

$$\frac{L_i}{\bar{L}} \cdot (1 + \sigma_i^2) \cdot \rho_i > \tau_{\mathrm{adaptive}}, \tag{7}$$

where L_i is the segment loss, $\bar{L}$ is the global average loss, σ_i^2 is the loss variance, ρ_i is the activation proportion, and τ_{adaptive} is the adaptive threshold updated during training. Candidate split points x_s are determined by maximizing a multi-criteria score, as shown in Equation 8:

$$S(x_s) = \alpha_1 E_{\mathrm{sep}}(x_s) + \alpha_2 C_{\mathrm{cons}}(x_s) + \alpha_3 B_{\mathrm{bal}}(x_s) + \alpha_4 Y_{\mathrm{sep}}(x_s), \tag{8}$$

where E_{sep} measures error separation, C_{cons} reflects internal consistency, B_{bal} evaluates data balance, Y_{sep} captures target separation, and $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ are trade-off weights.

Finally, segment-wise loss contributions are weighted according to morphology-aware importance, as shown in Equation 9:

$$w_i = 0.5\, I_{\mathrm{func}}(i) + 0.35\, I_{\mathrm{error}}(i) + 0.15\, D_i, \tag{9}$$

where $I_{\mathrm{func}}(i)$ denotes the functional importance of segment i, $I_{\mathrm{error}}(i)$ quantifies its error profile, and D_i represents data density. This weighting strategy guides training focus toward structurally and statistically critical segments.

In summary, ASNC, as a structural approximator, segmentally fits arbitrary nonlinear computations while keeping the output in BSE format, without requiring structural rewrites, and confines error injection strictly to a single nonlinear module. This mechanism not only prevents errors from diffusing along network depth and temporal dimensions but also preserves structural consistency throughout the forward path. On this basis, introducing lightweight Straight-Through Estimator (STE) training becomes safe and effective, since relying on equivalent BSE encoding and a consistent structural path ensures that errors are injected only at the nonlinear unit, thereby guaranteeing both training stability and final accuracy.

3.3 STE Training

With high-fidelity and structurally consistent forward propagation, we introduce the identity-gradient proxy STE, so that backpropagation only needs to approximate derivatives within an extremely small error space. Since the forward path is already very close to ANN behavior, the backward path only approximates locally. In this case, STE is no longer a "heuristic substitute" but a forward-consistent equivalent mapping, resulting in minimal error and stable convergence.

During the forward pass, the network operates on our defined hybrid strategy. For backpropagation, we construct a surrogate computational graph. The key lies in the gradient proxy chosen for the BSE encoder ($S_{\mathrm{out}} = \mathrm{EncodeBSE}(V_{\mathrm{in}})$). We employ a hard **Straight-Through Estimator (STE)**, defining its gradient as an identity mapping. This choice is **principled**, not merely heuristic, because our forward pass has already guaranteed extreme numerical fidelity through BSE and exact linear operations. The SNN's output is functionally consistent with its ANN counterpart—an assumption strongly supported by our end-to-end experiments (e.g., Table 8) where negligible performance degradation was observed even on 70B models. This empirical evidence allows us to safely assume the gradient can pass directly without more complex approximations. This leads to a more stable and efficient training process, grounding our methodology in established Quantization-Aware Training (QAT) theory.

In summary, although the entire network employs STE surrogate gradients during backpropagation, the first two steps (BSE and ASNC) have already ensured that the forward path remains highly equivalent and structurally consistent, thereby strictly confining error injection to the single nonlinear computation module. This design means that even with global use of STE, no additional approximation errors are introduced at multiple structural points; errors do not diffuse across depth or layers but remain concentrated, controllable, and easily traceable. Such a structure makes the training process more stable and the accuracy more reliable, while maintaining strong performance even under large-scale or short time-window conditions.

4 EXPERIMENTS

Our experiments systematically validate that our method can confine approximation error to a single controllable nonlinear module and maintain structural consistency of the forward and backward paths. On this basis, network training is more stable, more scalable, and exhibits significant energy-efficiency advantages. First, at the encoding level, under the same time-step budget, BSE yields errors about 10^{11} to 10^{18} times lower than rate coding and TTFS, demonstrating its effectiveness as a high-fidelity bidirectional mapping. Second, in system-level ablations, error is strictly confined to the ASNC unit, and the end-to-end deviation of the FFN is only 9.5×10^{-7}, verifying the error-localization mechanism. Next, at the end-to-end level, across MMLU, HellaSwag, ARC, and TruthfulQA, we retain no less than 98% of the ANN performance, with LLaMA-2 70B dropping by only 1.2% on MMLU and overall accuracy degradation under 2%. Furthermore, in large-model comparisons on LLaMA-2 7B, our perplexity is higher than the ANN baseline by only 0.46, which is 0.05% of the baseline method, reflecting the scalability afforded by structural consistency. Finally, at the hardware level, the energy consumption on Loihi 2 is about 2% of that on the GPU. These results collectively show that LASER achieves error localization and efficient scalability while preserving structural consistency.

4.1 VERIFICATION OF CORE COMPONENTS

4.1.1 FIDELITY AND COST OF BIT SPIKE ENCODING (BSE)

We first quantify reconstruction fidelity against latency cost under matched time-step budgets, comparing BSE with rate coding and TTFS. For BSE, we evaluate multiple precisions aligned with the step budget—INT2/INT4/FP8/FP16/FP32 at 2/4/8/16/32 steps, respectively; each evaluation uses 10,000 random values. For fairness, all FP16 experiments use 16 steps and all FP32 experiments use 32 steps. As shown in Table 1, BSE achieves an MSE of 1.03×10^{-9} at FP16 and 1.33×10^{-14} at FP32, approaching machine precision. By contrast, rate coding and TTFS under the same step budgets yield errors as high as 10^4–10^5. This means that BSE is 10^{11}–10^{18} times more accurate than the baselines. These results confirm that BSE provides a deterministic and reversible encoding, avoids the many-to-one information loss and error diffusion seen in traditional schemes, and establishes the foundation for confining error only to subsequent nonlinear modules.

4.2 ABLATION STUDIES

All ablations, unless otherwise stated, use Llama 2 7B on WikiText-2. We report two levels of evidence. At the *component level*, we decode SNN outputs and measure MSE against their FP16 ANN counterparts to identify where error originates. At the *system level*, we measure WikiText-2 perplexity (mean $\pm$ std over 5 seeds). The FP16 ANN yields a baseline PPL of 5.12.

4.2.1 COMPONENT-LEVEL RECONSTRUCTION FIDELITY

We probe the micro-behavior of BSE-based linear operations and ASNC-based nonlinearities by extracting a representative FFN layer, passing 1,024 random inputs through the FP16 reference, and comparing decoded SNN outputs via MSE. Table 2 shows that BSE-based linear operations achieve 1.45×10^{-9} MSE, essentially identical to the INT16 quantization floor (1.12×10^{-9}), while rate coding under the same setting degrades to 4.88×10^{-1}. The only visible error originates from ASNC, which is still bounded at 8.7×10^{-7} and over six orders of magnitude smaller than the rate-coded baseline. This confirms that error is strictly localized to the nonlinear module and that the linear path remains lossless under BSE.

4.2.2 LAYER-WISE SENSITIVITY AND PROGRESSIVE SNN-IZATION

We next quantify sensitivity by SNN-izing one subsystem at a time, then progressively composing them. Table 3 shows that FFN layers are the most sensitive, where replacing them alone increases PPL from 5.12 to 5.35 ($+0.23$). Attention layers show a smaller increase to 5.31, while Embedding and LayerNorm rise by only $+0.08$ and $+0.06$, respectively. Once all subsystems are converted, the final SNN reaches 5.58, a total increase of $+0.46$. This stepwise pattern confirms that errors add predictably and remain bounded rather than diffusing uncontrollably.

Table 2: Component-level reconstruction fidelity (MSE). Precision and the corresponding time-step budget (16 for FP16) are reported. BSE reaches the quantization floor in MSE, while ASNC confines error to 10^{-7}, over six orders smaller than rate coding. The result indicates that the mapping of BSE itself is high-fidelity and efficient.

SNN-ized Component	Core Technology	Precision	Time Steps	MSE
Quantized ANN (INT16)	Standard Quantization	FP16	—	1.12×10^{-9}
Linear Layer (SNN)	**BSE (Ours)**	FP16	16	$\mathbf{1.45 \times 10^{-9}}$
SiLU Activation (SNN)	**ASNC (Ours)**	FP16	16	$\mathbf{8.73 \times 10^{-7}}$
Full FFN Block (SNN)	**BSE + ASNC (Ours)**	FP16	16	$\mathbf{9.51 \times 10^{-7}}$
Linear Layer (SNN)	Rate Coding	FP16	16	4.88×10^{-1}

Table 3: Layer-wise SNN adoption impact on WikiText-2 perplexity. Precision and the corresponding time-step budget (16 for FP16) are given. FFNs increase PPL by $+0.23$, Attention by $+0.19$, while LayerNorm and Embedding remain nearly unchanged. The full model ends at $+0.46$. This result demonstrates that linear components remain precise, while approximation error is introduced only by nonlinear modules and is strictly localized, thereby validating the high-fidelity forward propagation of the entire network.

SNN-ized Component	Precision	Time Steps	Resulting PPL
None (FP16 Baseline)	FP16	—	5.12 ± 0.01
FFN Layers only	FP16	16	5.35 ± 0.03
Attention Layers only	FP16	16	5.31 ± 0.02
Embedding/Output only	FP16	16	5.20 ± 0.02
LayerNorm only	FP16	16	5.18 ± 0.01
Full SNN (All Components)	FP16	16	5.58 ± 0.04

As shown in Table 4, progressively adding sensitive subsystems makes the accumulation explicit. Starting from FFN-only at 5.35, adding the nonlinear Activation raises PPL to 5.45 ($+0.10$), which accounts for the major increase. Subsequently adding Attention only slightly increases PPL to 5.47 ($+0.02$). Incorporating Embedding raises it further to 5.55 ($+0.08$), and finally the Full SNN model results in 5.58 ($+0.03$). The saturation at less than half a point above baseline demonstrates that errors grow additively but remain tightly bounded, with the dominant contribution coming from the nonlinear Activation.

Table 4: Performance impact of progressive SNN-ization strategies. Precision and the corresponding time-step budget (16 for FP16) are reported. The cumulative error remains bounded: $+0.12$ from Activation, $+0.03$ from Attention, $+0.08$ from Embedding, and $+0.03$ from LayerNorm, totaling $+0.26$. This result indicates that BSE encoding is high-fidelity, with the main errors arising from the nonlinear component ASNC, while the overall system remains high-fidelity and structurally consistent.

SNN-ization Strategy (Progressive)	Precision	Time Steps	Resulting PPL
1. FFN only	FP16	16	5.35 ± 0.03
2. FFN + Activation	FP16	16	5.47 ± 0.04
3. FFN + Activation + Attention	FP16	16	5.50 ± 0.04
4. FFN + Activation + Attention + Embedding	FP16	16	5.58 ± 0.04
5. Full SNN (All Components)	FP16	16	5.61 ± 0.04

4.2.3 Fine-Grained Component Analysis

We further decompose the sensitive Attention and FFN modules. Within Attention, Table 5 indicates that SNN-izing the Q, K, V projections increases PPL by $+0.16$ to 5.28, whereas converting only the score path adds merely $+0.06$. This shows that linear projections dominate sensitivity, while other operations remain robust.

For FFN, Table 6 contrasts an *ideal* SNN ceiling that reuses the original `SiLU` with our **BSE+ASNC** design. The gap is only $+0.09$ PPL (5.18 vs. 5.21), quantifying the minor fidelity cost introduced by ASNC. In contrast, replacing ASNC with an inconsistent low-fidelity ReLU breaks structural consistency and degrades performance severely to 50.74 PPL, while replacing BSE with rate coding leads

Table 5: Fine-grained analysis of SNN-izing components within the Attention module. Precision and time-step budget (16 for FP16) are reported. Q/K/V projections raise PPL by $+0.16$, while the score path adds only $+0.06$, confirming that sensitivity is localized. This result indicates that the sensitivities of different linear computations vary slightly, but overall the BSE encoding is high-fidelity.

Attention Component SNN-ized	**Precision**	**Time Steps**	**Resulting PPL**
None (Baseline)	FP16	—	5.12 ± 0.01
QKV Proj. only	FP16	16	5.28 ± 0.03
Attn Score only	FP16	16	5.18 ± 0.02
Q+K only	FP16	16	5.24 ± 0.03
V only	FP16	16	5.22 ± 0.02
All Attention SNN	FP16	16	5.31 ± 0.05

to catastrophic failure beyond 100 PPL. These results demonstrate that BSE provides indispensable high-fidelity spike encoding, while ASNC ensures structural consistency of nonlinear mappings, and together they enable high-fidelity and structurally consistent forward propagation.

Table 6: Component contribution analysis in FFN layers. Results are reported with FP16 precision and a 16-step budget. ASNC adds only $+0.09$ PPL compared to the ideal ceiling, whereas replacing it with an inconsistent low-fidelity ReLU causes severe degradation to 50.74 PPL, and substituting BSE with rate coding leads to catastrophic failure beyond 100 PPL. These results show that the high-fidelity spike encoding provided by BSE and the structural consistency ensured by ASNC are both indispensable.

FFN Layer Configuration	**Precision**	**Time Steps**	**PPL**
Standard ANN with SiLU (FP16 Baseline)	FP16	—	5.12 ± 0.01
Ideal SNN (BSE + Standard SiLU)	FP16	16	5.18 ± 0.01
Full SNN (BSE + ASNC)	FP16	16	5.21 ± 0.02
SNN with Inconsistent Act. (BSE + ReLU)	FP16	16	50.74 ± 0.04
SNN with Lossy Encoding (Rate Coding + ASNC)	FP16	16	103.5 ± 10.4

4.2.4 Comparison with a Prior Baseline

As shown in Table 7, to contextualize our results we compare against a prior SNN baseline that evaluates spiking variants of LLaMA-2 7B on WikiText-2 under short time-step budgets. Our FP16 ANN baseline achieves a perplexity of 5.12, while our full SNN (BSE+ASNC, 16 steps) attains 5.58 ± 0.04, corresponding to only $+0.46$ higher than the ANN baseline. In contrast, the prior baseline reports perplexities between 11.85 and 14.16, i.e., $+6.7$ to $+9.0$ higher than the ANN baseline. Overall, compared to the prior baseline, our method reduces degradation by more than a factor of \$2.1\$–\$2.5\$, showcasing the significant fidelity advantage of LEASER over existing methods in large-scale modeling scenarios.

Table 7: Comparison on LLaMA-2 7B, WikiText-2 (lower PPL is better). Our method is only $+0.46$ above the ANN baseline, while a prior SNN method shows $+6.7$–$+9.0$ degradation, i.e., ours is 2.1–$2.5\times$ closer to the baseline. Results for ours are mean $\pm$ std over 5 seeds; baseline and prior are single-run as reported (no variance available). The experimental results show that the precise mapping provided by BSE encoding, together with the structural consistency ensured by ASNC, enables our method to achieve a clear fidelity advantage over existing approaches in large-scale modeling scenarios.

Method	**Quantization**	**Steps**	**PPL (WikiText-2)**
ANN Baseline	FP16	—	5.12
Ours (BSE+ASNC)	FP16	16	5.58 ± 0.04
SpikeLLM	W4A4	2	11.93
SpikeLLM	W4A4	4	11.85
SpikeLLM	W2A16	2	14.16
SpikeLLM	W2A16	4	14.16

4.3 End-to-End Performance and Scalability Analysis

We apply our framework to multiple open-source LLMs and report task accuracy under a fixed 16-step budget for SNNs. Table 8 shows that across MMLU, HellaSwag, ARC, and TruthfulQA, SNN performance consistently tracks the corresponding ANN baselines. For example, on LLaMA-2 70B, accuracy on MMLU drops by only 1.2% and the overall degradation across all benchmarks remains under 2%. On LLaMA-2 7B, perplexity rises by only $+0.46$ compared to the ANN, a negligible difference given the scale of the model. This result demonstrates that the component-level fidelity of BSE and localized error in ASNC carry through to full large-model systems, maintaining near-baseline accuracy under a short, fixed 16-step budget.

Table 8: End-to-end performance comparison (accuracy, %). Across five representative LLMs, accuracy degradation from ANN to SNN remains within 2%, with LLaMA-2 70B showing only -1.2% on MMLU. This confirms that fidelity holds at scale under a fixed 16-step budget. This result demonstrates that when extended to large-scale modeling scenarios, LEASER still maintains high-fidelity forward propagation.

Model	Steps	MMLU		HellaSwag		ARC		TruthfulQA	
		ANN	SNN	ANN	SNN	ANN	SNN	ANN	SNN
Phi-2 (2.7B)	16	58.11	56.0	75.11	72.5	61.09	58.9	44.47	42.1
Llama-2 (7B)	16	60.04	58.2	79.13	76.9	56.14	54.5	40.95	39.0
Mistral (7.3B)	16	60.78	59.1	84.88	83.0	63.14	61.5	68.26	66.0
Mistral ($8\times$7B)	16	68.59	68.0	86.03	85.2	67.24	66.5	59.54	58.3
Llama-2 (70B)	16	65.40	64.2	86.90	85.5	67.20	65.8	44.90	43.1

4.4 Empirical Analysis of Energy Efficiency on Neuromorphic Hardware

We finally examine energy in practice by benchmarking the core nonlinear component on Intel Loihi 2 and comparing against an ANN SiLU baseline on an NVIDIA A100 under matched precisions. Loihi 2 energy per operation is obtained from on-board probes. GPU energy per operation is estimated from sustained-load power measurements over millions of identical operations using `nvidia-smi`. As shown in Table 9, Loihi 2 consumes only about 0.5% of the GPU energy at both 16-bit and 32-bit settings, representing more than a $200\times$ reduction. Together with the earlier accuracy results showing less than 2% loss at scale, this demonstrates that the proposed design achieves substantial energy savings while preserving near-baseline accuracy, consistent with our roadmap toward precise, scalable SNNs.

Table 9: Relative energy efficiency of a single non-linear operation (ASNC vs. standard SiLU on GPU). Loihi 2 requires only 0.5% of GPU energy at both 16-bit and 32-bit, a more than $200\times$ saving, while preserving accuracy within 2% of the ANN baseline.

Precision	**Time Steps**	**Energy Ratio (Loihi 2 SNN / GPU ANN)**
16-bit	16	5.5×10^{-3}
32-bit	32	5.3×10^{-3}

5 Conclusion

LASER–SNN establishes a three–step roadmap: BSE guarantees strict equivalence in linear computations, ASNC confines approximation error to a single nonlinear module, and—on this high–fidelity, structurally consistent forward path—a lightweight identity–gradient STE enables stable and controllable training. Experimental results demonstrate that BSE achieves machine–level precision in encoding, ASNC introduces error only at one quantifiable point, and STE maintains stability without diffusion. On LLaMA–2 70B, accuracy on MMLU decreases by merely 1.2%, while on LLaMA–2 7B, perplexity increases by only +0.46. Combined with a $200\times$ improvement in energy efficiency on Loihi 2, these results show that LASER–SNN unifies fidelity, controllability, and scalability in large–scale spiking models.

REFERENCES

Daniel Auge, Julian Hille, Etienne Mueller, and Alois Knoll. A survey of encoding techniques for signal processing in spiking neural networks. *Neural Processing Letters*, 53(6):4693–4710, 2021. ISSN 1573-773X. doi: 10.1007/s11063-021-10562-2. URL https://doi.org/10.1007/s11063-021-10562-2.

Lina Bonilla, Jacques Gautrais, Simon Thorpe, and Timothée Masquelier. Analyzing time-to-first-spike coding schemes: A theoretical approach. *Frontiers in Neuroscience*, 16:971937, 2022. ISSN 1662-453X. doi: 10.3389/fnins.2022.971937. URL https://www.frontiersin.org/articles/10.3389/fnins.2022.971937.

Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, Yuyun Liao, Chit-Kwan Lin, Andrew Lines, Ruokun Liu, Deepak Mathaikutty, Steven McCoy, Arnab Paul, Jonathan Tse, Guruguhanathan Venkataramanan, Yi-Hsin Weng, Andreas Wild, Yoonseok Yang, and Hong Wang. Loihi: A neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359.

Mike Davies, Andreas Wild, Garrick Orchard, Yulia Sandamirskaya, Gabriel A. Fonseca Guerra, Prasad Joshi, Philipp Plank, and Sumedh R. Risbud. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–934, 2021. doi: 10.1109/JPROC.2021.3067593.

Shikuang Deng, Yuhang Li, Shanghang Zhang, and Shi Gu. Temporal efficient training of spiking neural network via gradient re-weighting, 2022. URL https://arxiv.org/abs/2202.11946.

Allen Gersho and Robert M. Gray. *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, Boston, MA, 1992. ISBN 978-0-7923-9181-4 / 0-7923-9181-0. doi: 10.1007/978-1-4615-3626-0.

Wenzhe Guo, Mohammed E. Fouda, Ahmed M. Eltawil, and Khaled Nabil Salama. Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems. *Frontiers in Neuroscience*, 15:638474, 2021. ISSN 1662-453X. doi: 10.3389/fnins.2021.638474. URL https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2021.638474.

Bobby He, Lorenzo Noci, Daniele Paliotta, Imanol Schlag, and Thomas Hofmann. Understanding and minimising outlier features in transformer training. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=npJQ6qS4bg.

Sangwoo Hwang, Seunghyun Lee, Dahoon Park, Donghun Lee, and Jaeha Kung. Spikedattention: Training-free and fully spike-driven transformer-to-SNN conversion with winner-oriented spike shift for softmax operation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=fs28jccJj5.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. Duquant: Distributing outliers via dual transformation makes stronger quantized LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=mp8u2Pcmqz.

Wolfgang Maass. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. ISSN 0893-6080. doi: 10.1016/S0893-6080(97)00011-7. URL https://www.sciencedirect.com/science/article/pii/S0893608097000117.

Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks, 2019. URL https://arxiv.org/abs/1901.09948.

Nitin Rathi and Kaushik Roy. Diet-snn: Direct input encoding with leakage and threshold optimization in deep spiking neural networks. *ArXiv*, abs/2008.03658, 2020. URL https://api.semanticscholar.org/CorpusID:221132082.

Nitin Rathi, Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation, 2020. URL https://arxiv.org/abs/2005.01807.

Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019. ISSN 1476-4687. doi: 10.1038/s41586-019-1677-2. URL https://doi.org/10.1038/s41586-019-1677-2.

Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017. ISSN 1662-453X. doi: 10.3389/fnins.2017.00682. URL https://www.frontiersin.org/articles/10.3389/fnins.2017.00682.

Sumit Bam Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time, 2018. URL https://arxiv.org/abs/1810.08646.

Ana Stanojevic, Stanisław Woźniak, Guillaume Bellec, Giovanni Cherubini, Angeliki Pantazi, and Wulfram Gerstner. High-performance deep spiking neural networks with 0.3 spikes per neuron, 2023. URL https://arxiv.org/abs/2306.08744.

Kaiwen Tang, Zhanglu Yan, and Weng-Fai Wong. Sorbet: A neuromorphic hardware-compatible transformer-based spiking language model, 2024. URL https://arxiv.org/abs/2409.15298.

X. Wang et al. Spatial-temporal self-attention for asynchronous spiking neural networks. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, 2023. URL https://www.ijcai.org/proceedings/2023/118.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12, May 2018. ISSN 1662-453X. doi: 10.3389/fnins.2018.00331. URL http://dx.doi.org/10.3389/fnins.2018.00331.

Tianlong Xing et al. A spiking large language model. *arXiv preprint arXiv:2410.11456*, 2025. URL https://arxiv.org/abs/2410.11456.

Xingrun Xing, Zheng Zhang, Ziyi Ni, Shitao Xiao, Yiming Ju, Siqi Fan, Yequan Wang, Jiajun Zhang, and Guoqi Li. Spikelm: Towards general spike-driven language modeling via elastic bi-spiking mechanisms, 2024. URL https://arxiv.org/abs/2406.03287.

Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer, 2023. URL https://arxiv.org/abs/2307.01694.

Kang You, Zekai Xu, Chen Nie, Zhijie Deng, Qinghai Guo, Xiang Wang, and Zhezhi He. Spikezip-tf: Conversion is all you need for transformer-based snn, 2024. URL https://arxiv.org/abs/2406.03470.

Lusen Zhao, Zihan Huang, Jianhao Ding, and Zhaofei Yu. TTFSFormer: A TTFS-based lossless conversion of spiking transformer. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=mJAa823xKu.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer, 2022. URL https://arxiv.org/abs/2209.15425.

Zhaokun Zhou, Kaiwei Che, Wei Fang, Keyu Tian, Yuesheng Zhu, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer v2: Join the high accuracy club on imagenet with an snn ticket, 2024. URL https://arxiv.org/abs/2401.02020.

Rui-Jie Zhu, Qihang Zhao, Guoqi Li, and Jason K. Eshraghian. Spikegpt: Generative pre-trained language model with spiking neural networks, 2024. URL https://arxiv.org/abs/2302.13939.

A ENTROPY PRESERVATION OF BSE

Setting. Fix a bit-window length $N \in \mathbb{N}$. For each channel, let the per-layer affine calibration be (λ, μ) with $\lambda > 0$ and $\mu \in \mathbb{R}$. Define the N-bit quantizer

$$\mathcal{Q}_N \triangleq \{0, 1, \ldots, 2^N - 1\}, \qquad q = \mathrm{clip}_{[0,\, 2^N-1]}\Big(\mathrm{round}\big(\lambda\,(x+\mu)\big)\Big). \tag{10}$$

The corresponding de-quantization (reconstruction) map is $\hat{x} = \lambda^{-1} q - \mu$. Write the binary expansion of q as

$$q = \sum_{t=0}^{N-1} b_t\, 2^{N-1-t}, \qquad b_t \in \{0, 1\}. \tag{11}$$

BSE encoder/decoder. Let the bit-weights and thresholds be the standard positional schedule

$$W_{\mathrm{bit}}(t) = 2^{\,N-1-t}, \qquad \Theta(t) = W_{\mathrm{bit}}(t), \qquad t = 0, \ldots, N-1. \tag{12}$$

Drive the non-leaky IF unit equation 1 with a single impulse carrying the integer code:

$$I_{\mathrm{in}}(t) = \delta_{t0}\, q, \qquad V_m(0) = 0. \tag{13}$$

The *BSE encoder* $\mathsf{E}_N : \mathcal{Q}_N \to \{0,1\}^N$ is defined as the spike train $S(t) \equiv S_{\mathrm{actual}}(t)$ produced by equation 1, $t = 0, \ldots, N-1$. The *BSE decoder* $\mathsf{D}_N : \{0,1\}^N \to \mathcal{Q}_N$ is the weighted sum

$$\mathsf{D}_N\big(\{s(t)\}_{t=0}^{N-1}\big) = \sum_{t=0}^{N-1} s(t)\, 2^{\,N-1-t}. \tag{14}$$

Lemma 1 (Bit-exact emission)**.** *Under equation 12–equation 13, the encoder emits the binary digits of q in MSB→LSB order:*

$$S(t) = b_t \quad \textit{for all } t = 0, \ldots, N-1,$$

and the membrane obeys $V_m(t) = \sum_{\tau>t} b_\tau\, 2^{\,N-1-\tau}$ for $t = 0, \ldots, N$ (with the empty sum 0 at $t = N$).

Proof. By equation 1 and equation 13,

$$V_m(0) = 0, \quad V_m(0) + I_{\mathrm{in}}(0) = q.$$

At $t = 0$ the threshold is $\Theta(0) = 2^{N-1}$. Hence $M_0 = \mathbb{1}\{q \geq 2^{N-1}\} = b_0$ and the post-update membrane is

$$V_m(1) = q - b_0\, 2^{N-1} = \sum_{\tau=1}^{N-1} b_\tau\, 2^{\,N-1-\tau}.$$

Assume the claim holds up to time t. Then $V_m(t) = \sum_{\tau>t} b_\tau\, 2^{N-1-\tau}$ and $\Theta(t) = 2^{N-1-t}$. Thus

$$M_t = \mathbb{1}\{V_m(t) \geq \Theta(t)\} = \mathbb{1}\Big\{ \sum_{\tau>t} b_\tau\, 2^{N-1-\tau} \;\geq\; 2^{N-1-t} \Big\} = b_t,$$

since the sum's leading term is $b_t\, 2^{N-1-t}$ and all lower-order terms are strictly smaller than 2^{N-1-t}. The update gives

$$V_m(t+1) = V_m(t) - M_t\, \Theta(t) = \sum_{\tau>t} b_\tau\, 2^{N-1-\tau} - b_t\, 2^{N-1-t} = \sum_{\tau>t+1} b_\tau\, 2^{N-1-\tau}.$$

By induction the statements hold for all t, and $V_m(N) = 0$. □

Theorem 1 (Encoder–decoder bijection)**.** *E_N and D_N are mutual inverses:*

$$\mathsf{D}_N\big(\mathsf{E}_N(q)\big) = q \quad \textit{for all } q \in \mathcal{Q}_N, \qquad \mathsf{E}_N\big(\mathsf{D}_N(S)\big) = S \quad \textit{for all } S \in \{0,1\}^N.$$

Proof. The first identity follows from Lemma 1 and equation 14:

$$\mathsf{D}_N\big(\mathsf{E}_N(q)\big) = \sum_{t=0}^{N-1} S(t)\, 2^{N-1-t} = \sum_{t=0}^{N-1} b_t\, 2^{N-1-t} = q.$$

For the second identity, any $S \in \{0,1\}^N$ defines $q^\star = \mathsf{D}_N(S)$ with binary digits $b_t^\star = S(t)$. Lemma 1 shows that driving the IF with $q^\star$ emits $b_t^\star$ at time t, i.e., $\mathsf{E}_N(q^\star) = S$. □

Theorem 2 (Entropy preservation at N-bit alignment)**.** *Let X be any real-valued random variable and let $Q = \mathrm{clip}\circ\mathrm{round}(\lambda(X+\mu)) \in \mathcal{Q}_N$ be its N-bit quantization equation 10. Let $S = \mathsf{E}_N(Q)$ be the BSE spike code under equation 12–equation 13. Then the (discrete) Shannon entropy is invariant:*

$$H(Q) = H(S).$$

Equivalently, the pushforward distribution on spike sequences is a relabeling of the code distribution.

Proof. By Theorem 1, E_N is a bijection between the finite sets $\mathcal{Q}_N$ and $\{0,1\}^N$. Hence, for all $s \in \{0,1\}^N$,

$$\mathbb{P}[S = s] = \mathbb{P}\big[Q = \mathsf{D}_N(s)\big].$$

Therefore

$$H(S) = -\sum_s \mathbb{P}[S=s]\log\mathbb{P}[S=s] = -\sum_q \mathbb{P}[Q=q]\log\mathbb{P}[Q=q] = H(Q).$$

□

Vector/batch case. For B samples and d channels, define $Q \in \mathcal{Q}_N^{B\times d}$ and $S \in \{0,1\}^{N\times B\times d}$ by applying (λ,μ) per channel and E_N elementwise. The map $\mathsf{E}_N^{\otimes Bd}$ is a bijection between the two finite sets, so the joint entropy is preserved:

$$H\big(Q_{1:B,1:d}\big) = H\big(S_{0:N-1,\,1:B,\,1:d}\big).$$

Exactness of spike-domain accumulation. Within a bit window, the Soma pre-accumulation equals decoding in equation 14:

$$Q_a \;=\; \sum_{t=0}^{N-1} S_a(t)\, W_{\mathrm{bit}}(t),$$

so for any weight matrix W the linear map $A = Q_a W^\top$ is identical to applying W to the decoded integers. Thus all spike-domain accumulations are *exact integer arithmetic*. The only approximation in the BSE pathway appears at the explicit rounding/clipping that defines Q (or Q_y in equation 3).

Corollary 1 (No error diffusion from BSE under N-bit alignment)**.** *Assume each layer uses (λ,μ) that matches its N-bit quantization grid and uses the binary schedule equation 12. Then, relative to the corresponding N-bit quantized ANN, the BSE forward is functionally identical: encoding, spike-domain accumulation, and decoding are exact and introduce* no additional error. *Consequently, any discrepancy from full precision equals the ANN's own N-bit quantization error and does not diffuse or amplify due to BSE.*

Remarks. (i) The proof only requires a positional system with unique representation; the binary schedule equation 12 is the canonical choice. (ii) Signed ranges are handled by the offset μ (cf. equation 3), which translates the dynamic range to $[0, 2^N - 1]$ before encoding and back after decoding; bijectivity and entropy preservation hold channelwise under this affine change of variables. (iii) The results extend verbatim to per-channel (λ_o, μ_o) and per-layer bit windows, provided N is the operative grid width used by the ANN baseline.

B PROOF OF ASNC ERROR BOUND SUPERIORITY

B.1 SETUP

Let $X \sim \mathcal{N}(0,1)$ and consider only the nonnegative half-axis $[0,\infty)$, which corresponds to the effective input of ReLU-type nonlinearities. We partition the input space into n intervals and approximate the nonlinear response by piecewise-constant reconstruction points $\{y_i\}_{i=1}^{n}$. The mean squared error (MSE) distortion is defined as

$$D_n = \mathbb{E}\big[(X - Q_n(X))^2\big].$$

B.2 NON-UNIFORM QUANTIZATION ERROR BOUND

According to Bennett's integral and Lloyd–Max theory (Gersho & Gray, 1992), for any continuous density f with finite second moment, the high-rate distortion of the optimal non-uniform quantizer is

$$D_n^{\text{non-uniform}} = \frac{1}{12}\, n^{-2} \left(\int_0^{\infty} f(x)^{1/3}\, dx \right)^3 + o(n^{-2}), \quad n \to \infty.$$

For Gaussian input $f(x)$, the integral is finite and well-defined, giving the asymptotic error constant $C_{\text{non-uniform}}$.

B.3 UNIFORM PARTITIONING ERROR BOUND (E.G., SPIKEZIP-FT)

Consider a uniform quantization scheme, such as SpikeZIP-FT (You et al., 2024), which partitions the truncated interval $[0, R]$ into n equal subintervals of width $\Delta = R/n$. The corresponding distortion is

$$D_n^{\text{uniform}} = \frac{1}{12}\, \Delta^2 \sum_{i=1}^{n} \Pr(X \in I_i) + \int_R^{\infty} (x - R)^2 f(x)\, dx.$$

Choosing $R = \Theta(\sqrt{\log n})$ makes the tail term negligible, yielding the asymptotic form

$$D_n^{\text{uniform}} = \frac{R^2}{12n^2} \int_0^{\infty} f(x)\, dx + o(n^{-2}).$$

This defines the uniform error constant C_{uniform}.

B.4 COMPARISON AND CONCLUSION

By Hölder's inequality, for any non-constant density f,

$$\Big(\int f^{1/3}(x)\, dx \Big)^3 < R^2 \int f(x)\, dx.$$

For Gaussian $f(x)$ this inequality is strict. Hence,

$$C_{\text{non-uniform}} < C_{\text{uniform}}.$$

Conclusion. For Gaussian inputs, ASNC (non-uniform partitioning) achieves a strictly smaller asymptotic error bound than uniform partitioning methods such as SpikeZIP-FT. Therefore, under the same number of intervals n, the error of ASNC satisfies

$$D_n^{\text{ASNC}} < D_n^{\text{SpikeZIP-FT}}.$$

This establishes the theoretical advantage of ASNC in terms of error–latency tradeoff.

C ASNC FORMAL SPECIFICATION

C.1 SEGMENTAL CODEC AND BSE RE-ENCODING

Core Objective. Each segmental codec $\mathcal{C}_i(x)$ utilizes a spiking neural structure:

$$\text{spikes}_i(t) = \text{LIF}\big(\text{quantize}(x\, s_i + b_i),\, \theta_i(t)\big), \tag{15}$$

$$\mathcal{C}_i(x) = \sum_{t=1}^{T} w_i(t)\, \text{spikes}_i(t), \tag{16}$$

where $w_i(t)$ are trainable decoding weights for segment i.

BSE-Compatible Output. The decoded analog value $\widehat{y}_i$ for each active segment is immediately re-encoded by the BSE encoder to produce $S_i^{\mathrm{BSE}}(t)$ within the same window T; the overall output is

$$S^{\mathrm{BSE}}(t) \;=\; \sum_{i=1}^{N} \mathcal{I}_i(x)\, S_i^{\mathrm{BSE}}(t). \tag{17}$$

C.2 Adaptive Splitting Algorithm

Split Criterion.

$$\frac{L_i}{\bar{L}} \cdot \left(1 + \sigma_i^2\right) \cdot \rho_i \;>\; \tau_{\mathrm{adaptive}}. \tag{18}$$

Adaptive Threshold.

$$\tau_{\mathrm{adaptive}} \;=\; \tau_{\mathrm{base}} \cdot \alpha_{\mathrm{stag}} \cdot \alpha_{\mathrm{prog}}, \tag{19}$$

where τ_{base} is a base threshold, and α factors correct for stagnation and training progress.

Stagnation Factor α_{stag}.

$$\alpha_{\mathrm{stag}} \;=\; \begin{cases} 0.2, & \text{if } N_{\mathrm{stag}}/N_{\mathrm{active}} > 0.6, \\ 0.4, & \text{if } N_{\mathrm{stag}}/N_{\mathrm{active}} > 0.3, \\ 1.0, & \text{otherwise.} \end{cases} \tag{20}$$

Progress Factor α_{prog}.

$$\alpha_{\mathrm{prog}} \;=\; \begin{cases} 0.4, & \text{if } (L_0 - L_t)/L_0 < 0.005, \\ 0.6, & \text{if } (L_0 - L_t)/L_0 < 0.02, \\ 1.0, & \text{otherwise.} \end{cases} \tag{21}$$

Split Point Selection.

$$S(x_s) \;=\; \alpha_1\, E_{\mathrm{sep}}(x_s) \;+\; \alpha_2\, C_{\mathrm{cons}}(x_s) \;+\; \alpha_3\, B_{\mathrm{bal}}(x_s) \;+\; \alpha_4\, Y_{\mathrm{sep}}(x_s). \tag{22}$$

C.3 Adaptive Weighting Mechanism

Morphology-Aware Weights.

$$w_i \;=\; 0.5\, I_{\mathrm{func}}(i) \;+\; 0.35\, I_{\mathrm{error}}(i) \;+\; 0.15\, D_i. \tag{23}$$

Regional Priority $R_{\mathrm{region}}(i)$.

$$R_{\mathrm{region}}(i) \;=\; \begin{cases} 3.0, & a_i < 0 \text{ and } b_i > 0 \quad \text{(Zero-Crossing Region)}, \\ 2.5, & b_i \le 0 \quad \text{(Purely Negative Region)}, \\ 2.0, & a_i \ge 0 \quad \text{(Purely Positive Region)}, \\ 1.0, & \text{otherwise.} \end{cases} \tag{24}$$