

## Copyright Undertaking

This thesis is protected by copyright, with all rights reserved.

**By reading and using the thesis, the reader understands and agrees to the following terms:**

1. The reader will abide by the rules and legal ordinances governing copyright regarding the use of the thesis.
2. The reader will use the thesis for the purpose of research or private study only and not for distribution or further reproduction or any other purpose.
3. The reader agrees to indemnify and hold the University harmless from and against any loss, damage, cost, liability or expenses arising from copyright infringement or unauthorized usage.

### IMPORTANT

If you have reasons to believe that any materials in this thesis are deemed not suitable to be distributed in this form, or a copyright owner having difficulty with the material being included in our database, please contact [lbsys@polyu.edu.hk](mailto:lbsys@polyu.edu.hk) providing details. The Library will look into your claim and consider taking remedial action upon receipt of the written requests.

ADVANCED SPIKING NEURON MODELS FOR  
ACCURATE AND ROBUST PATTERN  
RECOGNITION

XINYI CHEN

PhD

The Hong Kong Polytechnic University

2025

The Hong Kong Polytechnic University  
Department of Data Science and Artificial Intelligence

Advanced Spiking Neuron Models for Accurate and Robust  
Pattern Recognition

Xinyi Chen

A thesis submitted in partial fulfillment of the requirements for  
the degree of Doctor of Philosophy  
August 2025

## CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgment has been made in the text.

Signature: \_\_\_\_\_

Name of Student: Xinyi Chen



# Abstract

The human brain, the most advanced intelligent system in nature, demonstrates a remarkable ability to perform complex cognitive functions and adapt to dynamic environments with exceptional efficiency and robustness. Learning from the brain's computational principles and structural organization has long been a promising direction for advancing neuromorphic artificial systems. Spiking neural networks (SNNs), which emulate the brain's spike-driven computation, stand out as a promising approach due to their sparse, asynchronous, and robust information processing. At the core of SNNs are spiking neuron models, which go beyond the simple activation units used in traditional artificial neural networks (ANNs) by incorporating brain-inspired neuronal dynamics for spatiotemporal information processing. These properties render spiking neurons a compelling foundation for building accurate and robust pattern recognition systems.

Despite these advantages, the computational capabilities of current SNNs remain significantly limited compared to their biological counterparts. A fundamental bottleneck lies in the oversimplified design of spiking neuron models, which often abstract neurons as point-like units with minimal temporal dynamics. This simplification, aiming at balancing biological realism and computational efficiency, neglects the diverse neural coding schemes, complex internal structures, and rich neural modulation mechanisms observed in biological neurons, which collectively contribute to the brain's powerful pattern recognition capabilities.

To bridge this gap, this thesis proposes a brain-inspired framework for advancing spiking neuron models across four critical dimensions: information representation, structural modeling, neural modulation, and evaluation benchmarks. The main contributions are as follows:

First, to enrich information representation strategies of spiking neuron models, this thesis introduces a hybrid neural coding and learning framework. Taking inspiration from the rich diversity of neural coding strategies observed in biological systems, this framework comprises a “neural coding zoo” that integrates heterogeneous spiking neurons with diverse coding strategies and supports flexible layer-wise assignment to accommodate task-specific requirements. Moreover, a layer-wise learning method tailored for the effective training of hybrid coding schemes is introduced. Experiments on image classification and sound localization tasks demonstrate the superiority of the proposed framework in terms of accuracy, inference latency, energy efficiency, and noise robustness.

Second, to enhance the structural complexity of spiking neuron models, this thesis introduces the Parallel Multi-compartment Spiking Neuron (PMSN). Inspired by the compartmental organization of biological neurons, PMSN models each neuron as a set of interacting substructures, facilitating effective information exchange and integration across multiple timescales. Additionally, two parallelization strategies are proposed to accelerate training and improve scalability. Across a variety of pattern recognition tasks, PMSN consistently outperforms state-of-the-art spiking neuron models in both accuracy and training speed, while maintaining comparable computational cost.

Third, to incorporate biological neural modulation mechanisms into spiking neuron models, this thesis introduces TE-SNNs. Motivated by the temporal encoding in biological neurons, TE-SNNs integrate a brain-inspired Temporal Structure Encoding (TE) module that modulates neuronal activity in response to distinct input temporal structures. TE-SNNs exhibit powerful capabilities in modeling complex temporal

structures, achieving state-of-the-art performance across vision, speech, and text pattern recognition tasks. Their real-world applicability is further demonstrated in autonomous driving scenarios, where they produce optimal, robust, and efficient driving strategies under highly dynamic conditions.

Finally, to systematically track the research progress on spiking neuron models, this thesis introduces the Neuromorphic Sequential Arena (NSA), a comprehensive neuromorphic benchmark that offers an effective, versatile, and application-oriented evaluation framework. NSA encompasses seven real-world tasks derived from diverse neuromorphic application scenarios, each reflecting rich temporal dynamics across multiple timescales. It supports multi-dimensional evaluation across accuracy, training speed, memory footprint, and energy efficiency, providing a standardized and extensible platform for consistent comparison and future development.

Together, these contributions establish a biologically grounded and systematically structured roadmap for designing advanced spiking neuron models, significantly enhancing the accuracy and robustness of SNNs in complex pattern recognition tasks.

# Publications Arising from the Thesis

1. Xinyi Chen, Yinsong Yan, Hanwen Liu, Malu Zhang, Yujie Wu, Qinyuan Ren, Huajin Tang, Kay Chen Tan, and Jibin Wu, “Temporal Structure Encoding Drives Accurate and Robust Sensorimotor Processing in Spiking Neural Networks”, manuscript submitted to *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*.
2. Xinyi Chen, Jibin Wu, Chenxiang Ma, Yinsong Yan, Yujie Wu, and Kay Chen Tan, “PMSN: A Parallel Multi-compartment Spiking Neuron for Multi-scale Temporal Processing”, manuscript submitted to *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*.
3. Xinyi Chen, Qu Yang, Jibin Wu, Haizhou Li, and Kay Chen Tan, “A Hybrid Neural Coding Approach for Pattern Recognition with Spiking Neural Networks”, in *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 46, no. 5 (2024): 3064-3078.
4. Xinyi Chen, Chenxiang Ma, Yujie Wu, Kay Chen Tan, and Jibin Wu, “Neuromorphic Sequential Arena: A Benchmark for Neuromorphic Temporal Processing”, in *the 34th International Joint Conference on Artificial Intelligence (IJCAI)*, 2025. (Accepted).

5. Chenxiang Ma, Xinyi Chen, Yanchen Li, Qu Yang, Yujie Wu, Guoqi Li, Gang Pan, Huajin Tang, Kay Chen Tan, and Jibin Wu, “Spiking Neural Networks for Temporal Processing: Status Quo and Future Prospects”, manuscript submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*. (Co-first author).

# Acknowledgments

This work was done in the Machine Intelligence and Nature-inspired computing (MIND) Lab led by Professor Kay Chen Tan at the Department of Data Science and Artificial Intelligence, The Hong Kong Polytechnic University.

I am deeply grateful to my supervisors Professor Kay Chen Tan and Professor Jibin Wu. I am sincerely thankful for the inspiring academic environment and broad research vision they have provided, which laid the foundation for my doctoral journey. Their unwavering guidance and thoughtful feedback in the field of neuromorphic computing have been instrumental in shaping the direction and depth of my research. I am also deeply grateful for their warm encouragement and support in daily life, which made my PhD experience both productive and fulfilling. I would also like to thank Professor Yujie Wu, Professor Huajin Tang, and Professor Qinyuan Ren for their valuable advice and guidance.

I sincerely appreciate my colleagues at the MIND Lab and other collaborators. My heartfelt thanks go to Dr. Wu Lin and Dr. Zhenzhong Wang for their valuable guidance during the early stages of my PhD study. I am also grateful to Dr. Qu Yang from the National University of Singapore, whose broad knowledge in spiking neural networks greatly helped me get a quick and smooth start in my research. I would also like to thank Xun Zhou, Chenxiang Ma, Xiang Hao, Shimin Zhang, Yinsong Yan, Ziyuan Ye, Xianwei Chen, Yuhong Chou, Zeyang Song, and Yanchen Li for their academic support and the enjoyable collaborations we have shared. Finally, I

would like to thank all other members of the MIND Lab for creating a supportive and friendly environment throughout my PhD journey.

Last but not least, I would like to express my heartfelt gratitude to my family and friends for their continuous love and encouragement throughout this journey. Their constant support has been my greatest source of strength in times of challenge and doubt. I am especially grateful to my husband Wendi Ding, whose patience, understanding, and companionship have accompanied me every step of the way during my graduate studies.

# Table of Contents

|                                                       |              |
|-------------------------------------------------------|--------------|
| <b>Abstract</b>                                       | <b>i</b>     |
| <b>Publications Arising from the Thesis</b>           | <b>iv</b>    |
| <b>Acknowledgments</b>                                | <b>vi</b>    |
| <b>List of Figures</b>                                | <b>xiii</b>  |
| <b>List of Tables</b>                                 | <b>xvi</b>   |
| <b>List of Abbreviations</b>                          | <b>xviii</b> |
| <b>1 Introduction</b>                                 | <b>1</b>     |
| 1.1 Research Objectives and Contribution . . . . .    | 4            |
| 1.2 Thesis Organization . . . . .                     | 8            |
| <b>2 Literature Review</b>                            | <b>11</b>    |
| 2.1 Fundamentals of Spiking Neural Networks . . . . . | 11           |
| 2.2 Spiking Neuron Models . . . . .                   | 15           |

|          |                                                                                             |           |
|----------|---------------------------------------------------------------------------------------------|-----------|
| 2.2.1    | Spiking Neurons with Enhanced Expressive Power . . . . .                                    | 17        |
| 2.2.2    | Spiking Neurons with Enhanced Computational Efficiency . .                                  | 19        |
| 2.3      | Neural Coding Schemes . . . . .                                                             | 20        |
| 2.4      | Learning Algorithms for SNNs . . . . .                                                      | 23        |
| 2.5      | Neuromorphic Benchmarks . . . . .                                                           | 26        |
| <b>3</b> | <b>A Hybrid Neural Coding Approach for Pattern Recognition with Spiking Neural Networks</b> | <b>30</b> |
| 3.1      | Introduction . . . . .                                                                      | 31        |
| 3.2      | A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks . . . . .         | 35        |
| 3.2.1    | Neural Coding Zoo . . . . .                                                                 | 36        |
| 3.2.2    | Hybrid Coding Assignment Strategy . . . . .                                                 | 38        |
| 3.2.3    | Hybrid Coding Design for Image Classification . . . . .                                     | 39        |
| 3.2.4    | Hybrid Coding Design for Sound Localization . . . . .                                       | 42        |
| 3.2.5    | A Layer-wise Learning Method for Hybrid Coding . . . . .                                    | 44        |
| 3.3      | Experimental Results . . . . .                                                              | 48        |
| 3.3.1    | Superior Image Classification Performance . . . . .                                         | 48        |
| 3.3.2    | Effective and Rapid Training Convergence with Layer-wise Learning Method . . . . .          | 52        |
| 3.3.3    | Superior Energy Efficiency with Hybrid Coding . . . . .                                     | 54        |
| 3.3.4    | Precise, Rapid, and Robust Sound Localization . . . . .                                     | 57        |
| 3.3.5    | Ablation Study of Hybrid Coding Design . . . . .                                            | 59        |

|          |                                                                                                                  |           |
|----------|------------------------------------------------------------------------------------------------------------------|-----------|
| 3.4      | Discussion and Conclusion . . . . .                                                                              | 62        |
| <b>4</b> | <b>A Parallel Multi-compartment Spiking Neuron for Multi-scale Temporal Processing</b>                           | <b>64</b> |
| 4.1      | Introduction . . . . .                                                                                           | 65        |
| 4.2      | PMSN: A Parallel Multi-compartment Spiking Neuron . . . . .                                                      | 69        |
| 4.2.1    | Revisit: Single-compartment Spiking Neuron Model Struggles to Process Temporal Signals . . . . .                 | 70        |
| 4.2.2    | A Generalized Multi-compartment Spiking Neuron with Interactive Dynamics . . . . .                               | 71        |
| 4.2.3    | Efficient Parallel Computation of Temporal Dynamics . . . . .                                                    | 74        |
| 4.2.4    | Effective Temporal Gradient Propagation . . . . .                                                                | 79        |
| 4.3      | Experimental Results . . . . .                                                                                   | 80        |
| 4.3.1    | Effective Modeling of Temporal Dependencies across Distinct Timescales . . . . .                                 | 80        |
| 4.3.2    | Superior Accuracy in Static Image Classification . . . . .                                                       | 83        |
| 4.3.3    | Visualization of Multi-compartment Dynamics . . . . .                                                            | 84        |
| 4.3.4    | Simulation Acceleration . . . . .                                                                                | 86        |
| 4.3.5    | Computational Cost . . . . .                                                                                     | 87        |
| 4.3.6    | Ablation Study . . . . .                                                                                         | 89        |
| 4.4      | Discussion and Conclusion . . . . .                                                                              | 90        |
| <b>5</b> | <b>Temporal Structure Encoding Drives Accurate and Robust Sensorimotor Processing in Spiking Neural Networks</b> | <b>92</b> |

|          |                                                                                            |            |
|----------|--------------------------------------------------------------------------------------------|------------|
| 5.1      | Introduction . . . . .                                                                     | 93         |
| 5.2      | TE-SNN: A Spiking Neural Network Incorporated with Temporal Structure Encoding . . . . .   | 95         |
| 5.2.1    | Design of the TE Module . . . . .                                                          | 98         |
| 5.2.2    | Integration of TE Modulation Mechanism into Spiking Neuron Models . . . . .                | 99         |
| 5.2.3    | Effective Temporal Credit Assignment in TE-SNNs . . . . .                                  | 101        |
| 5.3      | Experimental Results . . . . .                                                             | 103        |
| 5.3.1    | TE-SNNs Enhance Temporal Structure Discrimination . . . . .                                | 103        |
| 5.3.2    | TE-SNNs Enhance Temporal Processing Involving Long-term Temporal Dependencies . . . . .    | 106        |
| 5.3.3    | TE-SNNs Improve Robustness against Device Non-idealities and Noise Perturbations . . . . . | 110        |
| 5.3.4    | Application of TE-SNNs in Autonomous Driving . . . . .                                     | 113        |
| 5.3.5    | Impact of TE Modulation Designs . . . . .                                                  | 116        |
| 5.3.6    | Experimental Setups . . . . .                                                              | 117        |
| 5.4      | Discussion and Conclusion . . . . .                                                        | 127        |
| <b>6</b> | <b>Neuromorphic Sequential Arena: A Benchmark for Neuromorphic Temporal Processing</b>     | <b>130</b> |
| 6.1      | Introduction . . . . .                                                                     | 131        |
| 6.2      | Neuromorphic Sequential Arena . . . . .                                                    | 133        |
| 6.2.1    | Design Principles . . . . .                                                                | 133        |

|          |                                                  |            |
|----------|--------------------------------------------------|------------|
| 6.2.2    | Tasks . . . . .                                  | 134        |
| 6.2.3    | Segregated Temporal Probe . . . . .              | 139        |
| 6.3      | Experimental Results . . . . .                   | 141        |
| 6.3.1    | Data Preprocessing . . . . .                     | 142        |
| 6.3.2    | Benchmark Effectiveness Validation . . . . .     | 144        |
| 6.3.3    | Performance Benchmarking of SNN Models . . . . . | 146        |
| 6.3.4    | Efficiency Benchmarking of SNN Models . . . . .  | 148        |
| 6.4      | Discussion and Conclusion . . . . .              | 153        |
| <b>7</b> | <b>Conclusion and Future Directions</b>          | <b>155</b> |
| 7.1      | Summary of Contributions . . . . .               | 156        |
| 7.2      | Future Work . . . . .                            | 158        |
|          | <b>References</b>                                | <b>162</b> |

# List of Figures

|     |                                                                                                                              |    |
|-----|------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1 | Thesis organization overview. . . . .                                                                                        | 10 |
| 2.1 | Comparison between biological neurons in the brain and spiking neurons in SNNs. . . . .                                      | 12 |
| 2.2 | Illustration of typical spiking neuron models, capturing varying levels of abstraction from real biological neurons. . . . . | 16 |
| 2.3 | Illustration of neural coding schemes observed in biological systems. .                                                      | 20 |
| 3.1 | Illustration of the proposed hybrid neural coding and learning framework in solving the image classification task. . . . .   | 33 |
| 3.2 | Comparison of computational performance and biological plausibility among diverse candidate neural coding schemes. . . . .   | 37 |
| 3.3 | Illustration of the phase encoding front-end used for the sound localization task. . . . .                                   | 42 |
| 3.4 | Test accuracy as a function of inference time on the CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets. . . . .                | 50 |
| 3.5 | Decision time distribution for the proposed hybrid coding models. . .                                                        | 51 |
| 3.6 | Learning dynamics of the proposed layer-wise learning methods. . . .                                                         | 53 |

|     |                                                                                                                                                                                 |     |
|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 3.7 | Comparison of localization precision under different SNRs among different SNN models. . . . .                                                                                   | 58  |
| 3.8 | The sound localization precision comparison under different noise conditions. . . . .                                                                                           | 61  |
| 4.1 | Comparison of neuronal structure and dynamics between the single-compartment model, biological neurons, and the proposed multi-compartment spiking neuron model. . . . .        | 68  |
| 4.2 | Illustration of the proposed PMSN model and its parallel implementation.                                                                                                        | 76  |
| 4.3 | The learning curves of PMSN models (with and w/o reset), PSN model families, and LIF models (with and w/o reset) on Sequential CIFAR-10 and Sequential CIFAR-100 tasks. . . . . | 82  |
| 4.4 | The visualization of PMSN model dynamics. . . . .                                                                                                                               | 85  |
| 4.5 | Comparison of simulation speed ratios among serial MSN with identical structure, LIF, PSN, SPSN, and PMSN. . . . .                                                              | 86  |
| 4.6 | Spike sparsity, accuracy, and empirical energy cost comparison among LIF, PSN families, and our proposed PMSN model across distinct model sizes. . . . .                        | 87  |
| 5.1 | Enhancing temporal processing capacities of SNNs through temporal structure encoding. . . . .                                                                                   | 97  |
| 5.2 | Evaluation of the temporal discrimination capability across three timing tasks. . . . .                                                                                         | 105 |
| 5.3 | Evaluation of long-term temporal dependency modeling capability using the Delayed Recall task. . . . .                                                                          | 107 |
| 5.4 | Loss landscape visualization. . . . .                                                                                                                                           | 108 |

|     |                                                                                         |     |
|-----|-----------------------------------------------------------------------------------------|-----|
| 5.5 | Evaluation of robustness against device non-idealities and noise perturbations. . . . . | 112 |
| 5.6 | Application of TE-SNNs in autonomous driving. . . . .                                   | 115 |
| 5.7 | FPGA-based neuromorphic hardware designed for TE-SNNs . . . . .                         | 125 |
| 6.1 | Comparison of the three training algorithms in STP. . . . .                             | 139 |

# List of Tables

|     |                                                                                                                                    |    |
|-----|------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.1 | Comparison of classification accuracy and inference time on the CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets. . . . .           | 49 |
| 3.2 | Comparison of energy efficiency with other SNN implementations when achieving a comparable classification accuracy. . . . .        | 56 |
| 3.3 | Comparison of sound localization accuracy, MAE, and inference time on the SLoClas dataset. . . . .                                 | 57 |
| 3.4 | Comparison of MAE, inference time, and SynOps among different neural coding combinations. . . . .                                  | 59 |
| 4.1 | Comparison of classification accuracy of different neuron models on long-term sequential tasks. . . . .                            | 81 |
| 4.2 | Comparison of different models in handling long-term temporal dependencies on Sequential CIFAR-10 and CIFAR-100 datasets. . . . .  | 82 |
| 4.3 | Comparison on ImageNet-1K Image Classification Dataset. . . . .                                                                    | 84 |
| 4.4 | Comparative analysis of theoretical energy cost among different neuron models with the same network structure. . . . .             | 88 |
| 4.5 | Ablation study of compartment number on spatiotemporal integration tasks Sequential CIFAR-10 and CIFAR-100 with $T = 32$ . . . . . | 89 |

|     |                                                                                                                                   |     |
|-----|-----------------------------------------------------------------------------------------------------------------------------------|-----|
| 5.1 | Comparison of model performance on real-world temporal processing benchmarks. . . . .                                             | 109 |
| 5.2 | Impact of TE modulation designs in sequential image classification tasks.                                                         | 116 |
| 5.3 | Comparison of vanilla and TE-modulated neuronal dynamics across different neuron models. . . . .                                  | 122 |
| 6.1 | Summary of task characteristics, evaluation metrics, and other configurations in the proposed NSA benchmark. . . . .              | 135 |
| 6.2 | Validating the effectiveness of static image recognition datasets in evaluating the temporal processing capacity of SNNs. . . . . | 145 |
| 6.3 | Validating the effectiveness of event-based vision datasets in evaluating the temporal processing capacity of SNNs. . . . .       | 145 |
| 6.4 | Validating the effectiveness of keyword spotting datasets in evaluating the temporal processing capacity of SNNs. . . . .         | 145 |
| 6.5 | Validating the effectiveness of NSA in evaluating the temporal processing capacity of SNNs. . . . .                               | 146 |
| 6.6 | Results of NSA benchmark for different spiking neuron models with SFNN or SRNN architectures. . . . .                             | 147 |
| 6.7 | Results of NSA benchmark for different neural architectures using LIF neurons. . . . .                                            | 148 |
| 6.8 | Comparison of SNN models in terms of the training speed, memory consumption, and energy efficiency. . . . .                       | 149 |
| 6.9 | Theoretical and empirical energy costs of different architectures and neurons. . . . .                                            | 152 |

# List of Abbreviations

| Abbrev. | Description                                   |
|---------|-----------------------------------------------|
| AC      | Accumulate                                    |
| AD      | Audio Denoising                               |
| AER     | Address-Event Representation                  |
| AHP     | After-Hyperpolarization                       |
| AI      | Artificial Intelligence                       |
| AL      | Autonomous Localization                       |
| ALIF    | Adaptive LIF                                  |
| ALR     | Automatic Lip-Reading                         |
| ANN     | Artificial Neural Network                     |
| ASR     | Automatic Speech Recognition                  |
| ASRNN   | SRNN with an Adaptive Threshold Mechanism     |
| BDETT   | Bioinspired Dynamic Energy-Temporal Threshold |
| BHRF    | Balanced Harmonic Resonate-and-Fire           |
| BPTT    | Back-Propagation Through Time                 |
| CE      | Cross-Entropy                                 |
| CE-LIF  | Context Embedding LIF                         |
| CER     | Character Error Rate                          |
| CLIF    | Complementary LIF                             |
| CQT     | Constant Q Transform                          |

| <b>Abbrev.</b> | <b>Description</b>                    |
|----------------|---------------------------------------|
| DEXAT          | Double Exponential Adaptive Threshold |
| DH-LIF         | Dendritic Heterogeneity LIF           |
| DVS            | Dynamic Vision Sensors                |
| EEG-MI         | Electroencephalogram Motor Imagery    |
| FFT            | Fast Fourier Transform                |
| FPGA           | Field-Programmable Gate Array         |
| GLIF           | Gated LIF                             |
| GSC            | Google Speech Commands                |
| GSN            | Gated Spiking Neuron                  |
| GSU            | Gated Spiking Unit                    |
| HAR            | Human Activities Recognition          |
| ICA            | Independent Component Analysis        |
| IF             | Integrate-and-Fire                    |
| I-LIF          | Integer LIF                           |
| ISI            | Inter-Spike Intervals                 |
| KLIF           | K-based LIF                           |
| LIF            | Leaky Integrate-and-Fire              |
| LLM            | Large Language Model                  |
| LSTM           | Long Short-Term Memory                |
| LTC            | Liquid Time Constant                  |
| LTD            | Long-Term Depression                  |
| LTI            | Linear Time-Invariant                 |
| LTL            | Local Tandem Learning                 |
| LTP            | Long-Term Potentiation                |
| MAC            | Multiply-and-Accumulate               |
| MAE            | Mean Absolute Error                   |
| MFCC           | Mel-Frequency Cepstral Coefficients   |

| Abbrev.  | Description                               |
|----------|-------------------------------------------|
| ML       | Machine Learning                          |
| MoE      | Mixture-of-Experts                        |
| MSE      | Mean Squared Error                        |
| MSO      | Medial Superior Olive                     |
| MTPC     | Multi-Tone Phase Coding                   |
| NC       | Neuromorphic Computing                    |
| N-DNS    | Neuromorphic Deep Noise Suppression       |
| NoTD     | No Temporal Domain                        |
| NSA      | Neuromorphic Sequential Arena             |
| PLIF     | Parametric LIF                            |
| PMSN     | Parallel Multi-Compartment Spiking Neuron |
| PPO      | Proximal Policy Optimization              |
| P-R      | Pinsky-Rinzel                             |
| PS-MNIST | Permuted Sequential MNIST                 |
| PSN      | Parallel Spiking Neuron                   |
| PSP      | Post-Synaptic Potential                   |
| PTB      | Penn Treebank                             |
| RADLIF   | Recurrent Adaptive LIF                    |
| RAPL     | Running Average Power Limit               |
| ReLU     | Rectified Linear Unit                     |
| RF       | Resonate-and-Fire                         |
| SDBP     | Spatial Domain Backpropagation            |
| SFA      | Spike Frequency Adaptation                |
| SFNN     | Feedforward SNN                           |
| SHD      | Spiking Heidelberg Digits                 |
| SI-SNR   | Scale-Invariant Signal-to-Noise Ratio     |
| S-MNIST  | Sequential MNIST                          |

| <b>Abbrev.</b> | <b>Description</b>                |
|----------------|-----------------------------------|
| SNN            | Spiking Neural Network            |
| SNR            | Signal-to-Noise Ratio             |
| SOTA           | State-Of-The-Art                  |
| SPSN           | Sliding Parallel Spiking Neuron   |
| SRM            | Spike Response Model              |
| SRNN           | Recurrent SNN                     |
| SSC            | Spiking Speech Commands           |
| SSL            | Sound Source Localization         |
| STBP           | Spatio-Temporal Backpropagation   |
| STC-LIF        | Spatio-Temporal Circuit LIF       |
| STDP           | Spike-Timing-Dependent Plasticity |
| STFT           | Short-Time Fourier Transform      |
| STP            | Segregated Temporal Probe         |
| SYNOPS         | Synaptic Operations               |
| TC-LIF         | Two Compartment LIF               |
| TCN            | Temporal Convolutional Network    |
| TDOA           | Time Difference Of Arrival        |
| TE             | Temporal Structure Encoding       |
| TE-SFNN        | Feedforward TE-SNN                |
| TE-SRNN        | Recurrent TE-SNN                  |
| TTFS           | Time-To-First-Spike               |
| ZOH            | Zero-Order Hold                   |

# Chapter 1

## Introduction

Humans possess an extraordinary capacity to recognize a face in milliseconds, discern emotional nuances in music, and rapidly interpret gesture sequences in dynamic environments, all with remarkably low energy consumption. These abilities exemplify pattern recognition [1], where systems extract and interpret information from sensorimotor signals and turn them into decisions. In artificial intelligence (AI), pattern recognition underpins tasks such as image classification, speech recognition, and the temporal processing of sequential data. Advances in AI systems over the past decade have facilitated remarkable progress in this domain, achieving near-human-level accuracy on standard datasets and robust perception in real-world environments. However, much of this progress has been driven by scaling up model size and training data, which in turn incurs enormous computational demands and high power consumption [2].

These rising demands have sparked growing interest in alternative approaches that emulate the brain’s remarkable computational capability and efficiency, which originates from the sophisticated computations performed by biological neurons — the fundamental processing units of the central nervous system [3]. Embedded within tightly organized networks, biological neurons communicate through discrete electri-

cal impulses known as “spikes”. These spikes are processed within intricate dendritic and synaptic structures, where inputs are combined in localized, nonlinear ways for spatial integration [4, 5]. At the same time, neurons exhibit rich temporal dynamics, such as fluctuations in membrane potential and excitability modulation, which enable them to integrate information over time [6]. Collectively, neurons function as powerful and efficient processors of complex spatiotemporal signals [7, 8].

In this context, neuromorphic computing (NC) has emerged as a concrete route to emulate neurobiological principles to design AI systems in silico [9, 10, 11]. By emulating the event-driven and asynchronous processing mechanisms of the brain, neuromorphic systems offer an alternative to conventional Von Neumann architectures and represent a practical path toward brain-inspired intelligence. Over the past decade, neuromorphic platforms have made substantial strides, leading to their successful deployment in edge applications where low power consumption and real-time responsiveness to dynamic environments are essential [12, 13, 14, 15, 16].

At the core of NC lie spiking neural networks (SNNs), which provide an algorithmic framework for implementing spike-based computation [17]. In contrast to traditional artificial neural networks (ANNs) that rely on continuous-valued activations and instantaneous weighted summation, SNNs simulate human brain behaviors by using spikes for information transmission and processing, with each spike digitally implemented as a binary event. This event-driven design supports inherently sparse and asynchronous processing, where neurons activate only upon spike firing, leading to high energy efficiency in both computation and memory access [18, 19]. Moreover, SNNs are inherently endowed with neuronal dynamics, which enable them to update internal states in response to incoming spikes and integrate temporal information over time. This intrinsic temporal dynamics make SNNs particularly well-suited for capturing temporal dependencies, a capability that is fundamental for pattern recognition tasks involving sequence data [20, 21]. Additionally, the discrete and threshold-driven nature of spike generation in SNNs contributes to improved robustness, as

---

they tend to be less sensitive to minor perturbations or background noise in input signals [22, 23]. Collectively, these properties position SNNs as a biologically plausible framework for building accurate and robust pattern recognition systems. While the biological principles embedded in SNNs offer promising advantages, they also introduce unique challenges in network optimization. In particular, the non-differentiable nature of spikes hinders the direct application of standard gradient-based learning methods. In response, direct training [24, 25, 26] and ANN-to-SNN conversion algorithms [27, 28, 29] have been widely explored. Concurrently, significant progress has been made in designing novel optimization techniques [30, 31, 32] and spiking architectures [33, 34, 35, 36], aiming to mitigate training instability and facilitate effective convergence. These advancements have significantly improved the expressiveness and scalability of SNNs. As a result, SNNs have shown increasing promise in a variety of pattern recognition tasks, including image classification [37, 38], speech processing [39, 40, 41], sound localization [42, 43, 44], biomedical time-series classification [45, 46, 47], and robotic control [48, 49, 50].

Despite recent promising progress, the design of spiking neuron models remains a critical bottleneck limiting the full potential of SNNs. As the fundamental computational unit, the spiking neuron model defines how neurons represent, integrate, and retain information. Consequently, the expressiveness, efficiency, and robustness of an SNN are intrinsically linked to the underlying spiking neuron model it employs. Despite their central role, most existing SNNs adopt highly simplified neuron models, such as the Leaky Integrate-and-Fire (LIF), which reduce biological neurons’ functional and structural complexity to a minimal abstraction to favor computational efficiency [51]. These models typically treat neurons as indivisible point-like units with uniform temporal dynamics, ignoring the diverse neural coding schemes, complex structural morphology, and rich neural modulation mechanisms observed in biological neurons. While computationally efficient, such simplifications restrict the network’s ability to model complex spatiotemporal patterns, establish long-term memory, and adapt ro-

bustly in dynamic environments [21]. To bridge the gap between biological realism and practical applicability, it is essential to revisit the design of spiking neuron models and explore new architectures that incorporate key principles from neuroscience while remaining efficient for scalable learning and deployment.

## 1.1 Research Objectives and Contribution

As existing progress in SNNs primarily relies on oversimplified neuron models that fail to capture the richness of neural computation, the design of spiking neuron models remains a critical bottleneck in achieving accurate and robust neuromorphic pattern recognition systems. To this end, this thesis identifies four key research gaps in current spiking neuron model design that remain underexplored:

- **Homogeneous Information Representation Strategies.** Information representation in spiking neurons is commonly realized through neural coding, which encodes sensorimotor signals in the spike domain via timing, rate, or patterns of spike events. State-of-the-art (SOTA) SNNs for pattern recognition tasks are primarily built upon homogeneous spiking neuron models that adopt a single, uniform neural coding scheme throughout the network. However, since each neural coding scheme possesses its own merits and drawbacks, these homogeneous coding SNNs lead to suboptimal trade-offs between accuracy, response time, efficiency, and robustness, all of which are crucial for practical applications. By contrast, biological neurons employ diverse coding schemes to represent multimodal information with both high fidelity and efficiency [52]. Motivated by this observation, this thesis underscores the need to develop hybrid frameworks that integrate heterogeneous neural coding schemes to achieve optimal performance across varied task conditions.
- **Oversimplified Internal Structures.** Biological neurons possess intricate morphological structures, such as dendritic trees and axonal branches, that facilitate so-

phisticated spatial and temporal integration across multiple scales [53]. In contrast, most spiking neuron models used in SNNs are implemented as point neurons and ignore the structural complexity found in biological neurons, limiting the computational capacity of single neurons. In the field of neuroscience, multi-compartment models offer a more faithful representation by dividing a neuron into interconnected subunits with interactions among them [54]. While these models show great potential in spatiotemporal processing, their computational cost makes them impractical for large-scale, real-time pattern recognition tasks. Therefore, there is a pressing need to develop efficient spiking neuron models that capture essential substructure interactions without incurring excessive computational cost.

- **Underexplored Neural Modulation Mechanisms.** Neural modulation plays a central role in biological cognition by dynamically regulating neuronal states in response to external, task-relevant contexts. In particular, biological neurons can efficiently perceive and respond to sensorimotor signals characterized by complex temporal structures, owing to their intrinsic ability to encode these structures and modulate neuronal dynamics accordingly [55, 56]. In contrast, the absence of these mechanisms in existing spiking neuron models hampers their performance on pattern recognition tasks requiring fine-grained temporal structure modeling, underscoring the necessity of incorporating neural modulation mechanisms into spiking neuron model designs.
- **Ineffective Evaluation Benchmarks.** While many novel approaches have been proposed, the lack of standardized and effective evaluation benchmarks has hindered consistent tracking and slowed progress in spiking neuron model development. Although existing SNN benchmarks have substantially advanced neuromorphic computing research, they often lack the rich temporal dependencies spanning multiple timescales that characterize real-world signals, reflect only a narrow scope of neuromorphic pattern recognition application scenarios, and fail to provide consistent evaluation metrics to assess model practicality. Consequently, they offer

limited insight into the true capabilities of spiking neuron models, particularly in temporal processing. This highlights the urgent need for a comprehensive benchmark that captures diverse real-world conditions and provides effective assessments of model performance.

Building on the limitations identified above, this thesis advances the design of spiking neuron models to overcome these challenges, thereby enhancing the accuracy and robustness of SNNs across diverse pattern recognition tasks. To this end, the thesis presents a systematic framework that integrates brain-inspired principles, including enriched information representation strategies, enhanced structural complexity, and novel neural modulation mechanisms, into spiking neuron designs in a computationally efficient manner. In addition, it introduces a comprehensive evaluation framework that enables effective, consistent benchmarking and facilitates seamless integration of emerging spiking neuron models. The main contributions of this thesis are summarized as follows:

First, to enrich information representation strategies of spiking neuron models, this thesis introduces a hybrid neural coding and learning framework, which integrates heterogeneous coding schemes to enhance the fidelity, efficiency, robustness, and adaptability of neural representations. Drawing on the biological diversity of neural coding, the framework encompasses a neural coding zoo with diverse neural coding schemes discovered in neuroscience, together with a flexible assignment strategy that allocates different coding schemes across layers based on task demands. In addition, layer-wise learning methods are developed to facilitate effective training of hybrid coding SNNs. Experiments demonstrate the superiority of the proposed framework on image classification and sound localization tasks. Specifically, the hybrid coding SNNs achieve comparable accuracy to SOTA SNNs, while exhibiting favorable trade-offs among inference latency, energy consumption, and noise robustness. This study yields valuable insights into hybrid neuron model designs, laying the foundation for constructing more accurate, robust, and efficient SNNs.

Second, to enhance the structural complexity of spiking neuron models, this thesis introduces a Parallel Multi-compartment Spiking Neuron (PMSN), which incorporates the intricate internal structure of biological neurons to enhance model expressiveness while retaining computational efficiency. Inspired by the compartmental organization of biological neurons, PMSN incorporates multiple interactive substructures that enable effective information exchange and integration across diverse timescales. Additionally, to address the computational burden associated with the increased complexity of the proposed model, this thesis introduces two parallelization techniques that decouple the temporal dependencies of neuronal dynamics, allowing parallelized computation across time. The experimental results on a wide range of pattern recognition tasks demonstrate the superiority of PMSN. It outperforms existing spiking neuron models in terms of multiscale temporal processing capacity and training speed, while maintaining comparable computational cost. Specifically, compared with the commonly used Leaky Integrate-and-Fire neuron, PMSN offers a simulation acceleration of over  $10\times$  and a 30% improvement in accuracy on the Sequential CIFAR-10 dataset, while maintaining comparable computational cost. Therefore, the proposed PMSN presents a promising solution to harness the computational advantages of detailed neuron models in NC.

Third, to incorporate biological neural modulation mechanisms into spiking neuron models, this thesis introduces TE-SNNs, which integrate a brain-inspired Temporal structure Encoding (TE) module to modulate neuronal dynamics, thereby enhancing their temporal processing capability and robustness. Inspired by temporal encoding in biological neurons, the TE module encodes task-specific temporal structures and modulates neuronal dynamics to generate bursting spikes when inputs align with the encoded patterns. TE-SNNs not only excel in timing tasks explored in neurocognitive experiments, such as interval, duration, and periodicity discrimination, but also demonstrate significantly improved accuracy and robustness in modeling complex temporal structures in visual, speech, and textual signals. Furthermore, we demon-

strate the practical applicability of TE-SNNs in an autonomous driving task, where they generate optimized driving strategies while achieving substantial improvements in energy efficiency and robustness, showcasing the potential of deploying advanced spiking neuron models in complex, resource-constrained real-world environments.

Finally, to better track the progress and guide the advancement of spiking neuron models in pattern recognition, this thesis introduces the Neuromorphic Sequential Arena (NSA), a comprehensive benchmark designed to systematically evaluate the temporal processing capacities of SNNs across diverse application scenarios. The NSA includes seven real-world tasks, each capturing rich temporal dynamics across multiple timescales. Utilizing NSA, this thesis conducts extensive comparisons of recently introduced spiking neuron models and neural architectures, presenting a comprehensive evaluation in terms of accuracy, training speed, memory usage, and energy efficiency. Our findings highlight the pressing need for spiking neuron models that can maintain both accuracy and efficiency across tasks with diverse temporal dynamics. By offering an effective, versatile, and application-oriented evaluation framework, NSA paves the way toward the systematic development of accurate and robust neuromorphic pattern recognition systems.

## 1.2 Thesis Organization

This thesis consists of seven chapters:

- Chapter 1 introduces the background, motivation, and scope of this thesis. It outlines the key challenges in current spiking neuron model design and presents the overall research objectives and major contributions.
- Chapter 2 provides a comprehensive overview of SNNs. It first introduces the fundamentals of SNNs, followed by an in-depth discussion of spiking neuron models, along with their neural coding schemes and training methodologies. Finally, it

reviews existing neuromorphic benchmarking practices. This chapter establishes the technical foundation for understanding the key topics in subsequent chapters.

- Chapter 3 proposes a hybrid neural coding and learning framework. It introduces a neural coding zoo encompassing diverse biological neural coding schemes, a flexible coding assignment strategy, and layer-wise learning methods. Its effectiveness is validated through extensive experiments on image classification and sound localization tasks.
- Chapter 4 introduces a brain-inspired multi-compartment spiking neuron model. It begins with a generalized formulation of multi-compartment models, followed by detailed descriptions of neuronal dynamics, discretization methods, and two parallel computation strategies. The model’s spatiotemporal modeling capacity is then validated on diverse pattern recognition tasks, alongside an empirical analysis of its computational efficiency.
- Chapter 5 presents a novel neural modulation mechanism that leverages a brain-inspired TE module to modulate neuronal dynamics. This chapter details the design of the TE module and how it regulates spiking neuronal dynamics, followed by evaluations on various temporal discrimination and pattern recognition tasks. Finally, its practical value is demonstrated through a real-world autonomous driving application.
- Chapter 6 presents a novel neuromorphic benchmark framework, outlining its design principles, task definitions, and an evaluation tool for assessing benchmark effectiveness. Based on this framework, a comprehensive comparative evaluation is conducted across existing spiking neuron models and neural architectures.
- Chapter 7 summarizes the key findings and contributions of the thesis. It discusses the current limitations and outlines future research directions for the development of neuromorphic pattern recognition systems.

Figure 1.1 presents an overview of the thesis organization, highlighting the logical flow and illustrating the relationships between chapters.

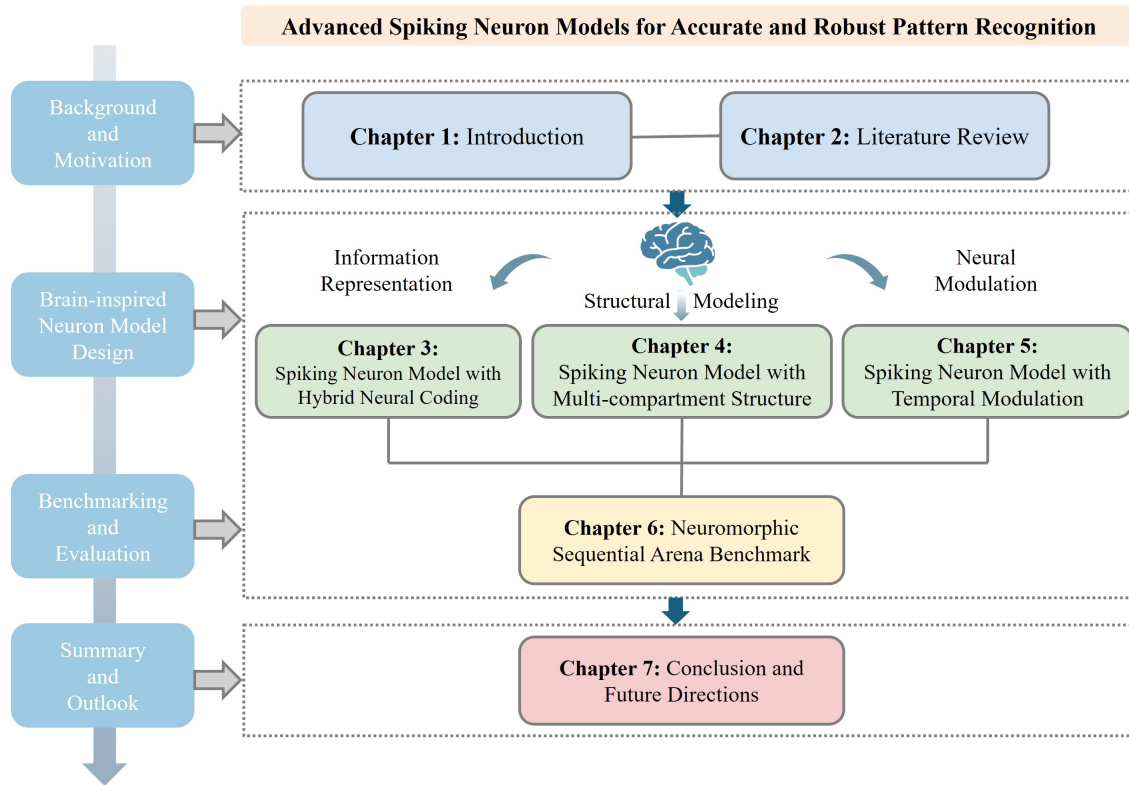


**Figure 1.1: Thesis organization overview**

# Chapter 2

## Literature Review

Inspired by the sparse and event-driven computational principles of biological neurons, SNNs have received growing attention over the past decades for their potential for accurate, robust, and energy-efficient computing. This chapter presents a systematic review of recent advances in SNNs, with a particular focus on the design of spiking neurons, along with their underlying neural coding schemes and training methodologies. We begin by introducing the fundamentals of SNNs, followed by a comprehensive overview of the evolution of spiking neuron models. Next, we examine neural coding schemes observed in biological systems and their implementations in SNNs, and then review the learning algorithms developed to address the discrete and non-differentiable nature of spike-based dynamics. Finally, we discuss existing neuromorphic benchmarks and highlight the challenges in evaluating spiking neuron performance, thereby laying the foundation for the contributions of this thesis.

### 2.1 Fundamentals of Spiking Neural Networks

The human brain, with its remarkable ability to process complex and noisy sensory signals in real time, has long inspired the design of neuromorphic computing systems.

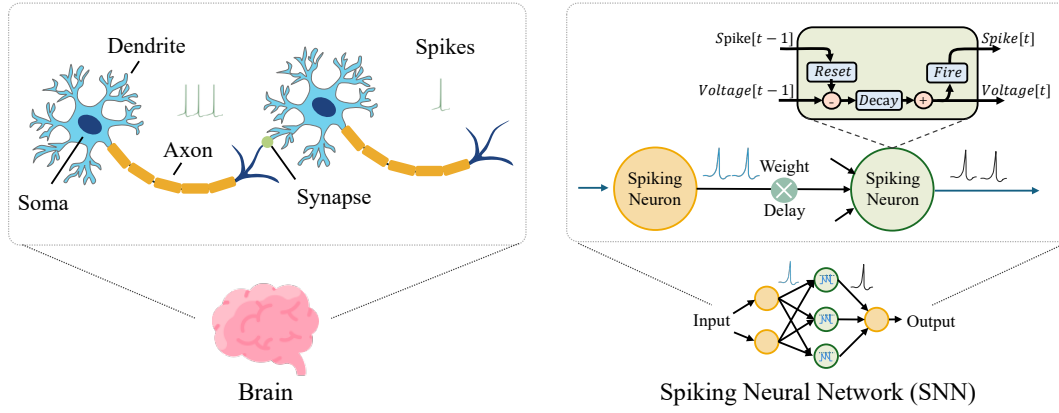


**Figure 2.1: Comparison between biological neurons in the brain and spiking neurons in SNNs.** Adapted from [57].

As illustrated in Figure 2.1, the design of SNNs is inspired by the structure and function of biological neurons [17]. A typical neuron consists of dendrites, a soma, and an axon, exhibiting rich temporal dynamics while communicating with other neurons via spikes through synaptic connections. In SNNs, these biological principles are abstracted into computational models that consist of three fundamental components:

- **Neuronal dynamics**, which govern how the internal states of neurons evolve over time in response to inputs [3, 58];
- **Neural coding schemes**, which define how information is represented and transmitted through the spatiotemporal patterns of spikes;
- **Synaptic connectivity**, which determines the topology and functional properties of connections between neurons.

Specifically, neuronal dynamics describe the biophysical processes by which neurons integrate synaptic inputs, adapt internal states over time, and generate spikes. To approximate these processes in a computationally efficient manner, the LIF [51] model has become the most widely used spiking neuron model due to its balance between

simplicity and biological plausibility. The LIF model abstracts away the rich structural and dynamic complexity of biological neurons, and instead treats the neuron as a point-like, indivisible unit with homogeneous dynamics. It maintains a membrane potential  $V(t)$ , which integrates incoming current and decays over time. Its subthreshold behavior is governed by the following first-order differential equation:

$$\tau_m \frac{dV(t)}{dt} = -(V(t) - V_{rest}) + RI(t), \quad (2.1)$$

$$\text{if } V(t) \geq \theta, S(t) = 1, V(t) \rightarrow V_{rest}; \text{ else } S(t) = 0. \quad (2.2)$$

where  $\tau_m$  is the time constant that governs the rate at which the membrane potential decays,  $V_{rest}$  is the resting potential,  $R$  is the membrane resistance,  $I(t)$  is the input current, and  $S(t)$  is the output spike. When the membrane potential exceeds a predefined threshold  $\theta$ , a spike is generated, followed by a reset of the membrane potential to  $V_{rest}$ . To facilitate efficient simulation on modern machine learning (ML) frameworks, the LIF model is typically discretized using Euler methods [59]. This results in the following discrete-time update rule:

$$V[t] = \alpha V[t-1] (1 - S[t-1]) + RI[t], \quad (2.3)$$

$$S[t] = H(V[t] - \theta), \quad (2.4)$$

where  $\alpha = \exp(-\frac{1}{\tau_m})$  is the decay factor and  $H(\cdot)$  denotes the Heaviside step function. The reset is implemented by the term  $(1 - S[t-1])$ . Despite its computational simplicity and ease of hardware deployment, the LIF model falls short in capturing the intricacies of biological neurons. Moreover, its limited expressiveness constrains its ability to model complex spatiotemporal dependencies, and its slow training speed hinders scalability to long-sequence tasks. In Section 2.2, we provide a systematic review of spiking neuron models, from biologically grounded variants to recent extensions that enhance computational expressiveness and efficiency.

Neural coding schemes mimic the efficient and robust strategies employed by biological neurons to represent and transmit information via spike trains, such as spike rates or

precise spike timing. In SNNs, these schemes are typically implemented by defining how signals are encoded into and decoded from the spatiotemporal patterns of spike trains, which in turn requires designing specific neuron models whose integration and firing mechanisms are tailored to support the chosen coding scheme. Therefore, in this thesis, the enhancement of neural coding schemes is treated as a core element in the development of advanced spiking neuron models, which significantly improve the trainability, latency, robustness, and energy efficiency of SNNs. We provide a detailed review of representative neural coding schemes and their modeling approaches in Section 2.3.

Synaptic connectivity models the spatial and temporal structure of inter-neuronal spike propagation, wherein spikes from presynaptic neurons influence postsynaptic targets with varying synaptic strengths and transmission delays. In SNNs, these connections are typically abstracted into learnable synaptic weights and delays. For the  $j^{th}$  neuron in layer  $l$ , its input current can be written as:

$$I_j^l[t] = \sum_i W_{ij}^l S_i^{l-1}[t - d_{ij}^l], \quad (2.5)$$

where  $W_{ij}^l$  denotes the synaptic weight, which quantifies the connection strength from presynaptic neuron  $i$  to postsynaptic neuron  $j$ .  $d_{ij}^l$  represents the synaptic delay, modeling the time it takes for a spike to travel from the presynaptic neuron  $i$  to the postsynaptic neuron  $j$ , capturing the non-instantaneous nature of biological spike propagation. By stacking all neuron states in layer  $l$  into vector form (denoted in **bold**) and representing connectivity via the synaptic weight matrix  $\mathcal{W}^l$  and delay matrix  $\mathcal{D}^l$ , the layer-wise spike propagation in an SNN can be compactly expressed as:

$$\mathbf{I}^l[t] = \mathcal{W}^l \mathbf{S}^{l-1}[t - \mathcal{D}^l], \quad (2.6)$$

Together with the neuron's intrinsic dynamics and neural coding strategies, synaptic connectivity enables SNNs to perform complex and robust computations in a biologically inspired yet computationally efficient manner. However, the inherent non-

differentiability of the spike function  $H(\cdot)$  poses a significant challenge for optimizing synaptic parameters using standard gradient-based methods. This limitation has motivated the development of specialized training algorithms, which are introduced in Section 2.4.

## 2.2 Spiking Neuron Models

The fundamental computational units of the central nervous system are neurons, which possess highly intricate internal structures and neuronal dynamics that enable complex information processing. To mimic the computational principles underlying biological intelligence, artificial neurons have been extensively studied in the field of AI. Tracing back to 1943, McCulloch and Pitts proposed the first mathematical model of a neuron [60], which abstracted neural activity as a simple thresholding operation over the weighted sum of binary inputs. Building on this, Rosenblatt introduced the Perceptron [61] one decade later, a model capable of approximating nonlinear input-output mappings through the optimization of synaptic weights and thresholds, laying the groundwork for the modern neural network framework. However, biological neurons are far more than binary switches. They are sophisticated computational units that integrate synaptic inputs over space and time, modulate their internal states dynamically, and emit spike trains that drive neural activity across the brain. These rich dynamics arise from the interplay among diverse neural coding schemes, complex morphological structures, intricate gated ionic dynamics, and rich neural modulation mechanisms. Accordingly, designing advanced spiking neuron models that capture biological details with computational efficiency is crucial for unlocking the full potential of SNNs. This section reviews the recent progress in spiking neuron model designs, highlighting key innovations in neuronal dynamics and structures.

Modern research on spiking neuron models builds upon the seminal work of Hodgkin and Huxley in 1952 [63], which described ionic conductances and action potential gen-

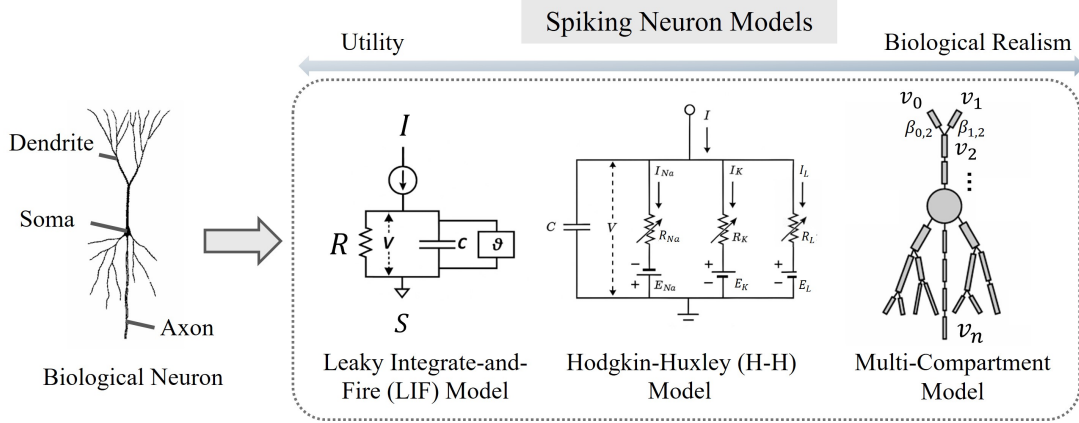


**Figure 2.2:** Illustration of typical spiking neuron models, capturing varying levels of abstraction from real biological neurons. Adapted from [62].

eration using coupled nonlinear differential equations, marking a foundational step in computational neuroscience. Subsequently, Rall’s compartmental framework [64] extended biological neuron modeling by introducing multi-compartment structures, allowing simulations of signal propagation across dendrites and soma to capture the structural complexity of biological neurons. While these models offer high biological fidelity, their large state spaces and computational cost render them impractical for large-scale neural simulations. Therefore, more tractable models such as the Izhikevich neuron [65] and Resonate-and-Fire (RF) neuron [66] were proposed, reducing the system to two coupled variables while preserving key dynamical behaviors like bursting and resonance. These models offer a better trade-off between biological realism and computational tractability. However, their complex nonlinear dynamics can still pose challenges for optimization and scalability in deep SNNs.

As a result, the SNN community has long relied on simplified abstractions to facilitate scalable and efficient learning. Among these, the non-leaky Integrate-and-Fire (IF) and LIF models [51] are the most widely adopted, in which neurons are reduced to point-like units with only minimal temporal dynamics. The Spike Response Model (SRM) [67] further generalizes the LIF model by expressing membrane potential updates as a convolution over presynaptic input history, enabling efficient temporal

integration. Furthermore, extensions such as Parametric LIF (PLIF) [68], K-based LIF (KLIF) [69], and Gated LIF (GLIF) [70] introduce learnable time constants, spike scaling factors, and dynamic gating factors into the LIF model to enhance its learning ability. While these simplified models offer ease of implementation and have become the default choice in many SNN frameworks, they often fall short in two critical aspects: (1) These point models discard important spatial structure and temporal dynamic complexity of biological neurons, limiting their expressiveness to model rich spatiotemporal dependencies; (2) When trained using gradient-based algorithms such as Back-Propagation Through Time (BPTT), their sequential computation across time leads to slow training, posing significant challenges for scaling to long sequence modeling tasks.

To address these limitations, recent research has advanced spiking neuron models along two major directions: enhancing expressive power through structural and temporal complexity, and improving computational efficiency via parallelizable architectural designs. We review these advancements in detail below.

### 2.2.1 Spiking Neurons with Enhanced Expressive Power

To enhance the expressiveness of spiking neuron models, researchers have made advances in three main directions:

**Dynamic Diversity.** Some studies extend the simplified LIF model by expanding neuronal states and incorporating diverse brain-inspired dynamics. For example, to mimic the firing rate adaptation observed in biological neurons, models such as Adaptive LIF (ALIF) [46], Double Exponential Adaptive Threshold (DEXAT) [71], Spike Frequency Adaptation (SFA) [72], and Bioinspired Dynamic Energy-Temporal Threshold (BDETT) [49] extend the LIF neuron with dynamic firing thresholds to model threshold adaptation mechanisms, while the After-Hyperpolarization (AHP) model [73] introduces an additional state to capture after-hyperpolarizing currents.

In this line, Liquid Time Constant (LTC) neuron [74] further integrates gating mechanisms into membrane potential integration and threshold adaptation. Moreover, Recurrent adaptive LIF (RadLIF) [20] and Complementary LIF (CLIF) [75] models integrate biological features, including adaptation currents and complementary membrane potentials, into neuronal dynamics. These designs expand the representational space of spiking neurons and have demonstrated effectiveness in tasks such as image recognition and speech processing, although their ability to capture more complex long-term dependencies remains limited.

**Structural Complexity.** Inspired by the rich morphological structure of biological neurons, another line of work introduces structurally enhanced spiking models with higher biological realism and computational power. Rather than treating the neuron as an indivisible unit with homogeneous dynamics, these models decompose single neurons into multiple substructures with heterogeneous dynamics. For instance, Dendritic Heterogeneity LIF (DH-LIF) [47] introduces heterogeneous dendritic branches to capture multiscale temporal features, Spatio-Temporal Circuit LIF (STC-LIF) [76] embeds intra-neuronal feedback loops within the neuronal structure to enable state modulation, and Two Compartment LIF (TC-LIF) [21] explicitly captures soma-dendrite interactions to enable more complex temporal integration. These structurally enhanced models have shown promising results on real-world pattern recognition tasks such as speech recognition and biological signal classification. However, their improved expressiveness often comes at the cost of increased computational cost, posing practical challenges for large-scale implementation.

**Propagation Delays.** Recent studies have further extended spiking neuron models by incorporating trainable temporal delays into spike propagation [77, 78]. These models capture the biological reality that signal conduction along axons and dendrites is not instantaneous. By incorporating these delays into neuronal dynamics, neurons gain enhanced capability for short-term temporal feature extraction, which is critical for pattern recognition tasks such as speech recognition.

### 2.2.2 Spiking Neurons with Enhanced Computational Efficiency

Computational efficiency remains a crucial bottleneck for applying SNNs to large-scale pattern recognition tasks. A central challenge lies in the inherently sequential nature of neuronal state updates, whereby training time grows linearly with sequence length, posing significant challenges for handling long sequences. A promising direction is to reformulate the neuronal state update equations into linear time-invariant forms, enabling efficient parallel computation across time. Specifically, in contrast to traditional spiking neurons that require sequential step-by-step computation, these novel formulations allow the entire sequence of temporal dynamics to be processed in parallel on modern GPU-based frameworks, significantly accelerating training and inference speed. One early effort in this vein is the family of Parallel Spiking Neuron (PSN) models [79], which restructure membrane dynamics into a temporal matrix multiplication form, thereby significantly improving training throughput. In addition, the Integer LIF (I-LIF) model [80] adopts integer-valued spikes to enhance representational capacity during parallel training, while retaining the serial, event-driven processing nature during inference to ensure deployment feasibility.

Building upon these advancements, this thesis further enhances the existing spiking neurons by introducing enhanced structural modeling, enriched neuronal dynamics, and novel neuromodulation mechanisms. Specifically, in Chapter 4, we propose a novel structure-enhanced multi-compartment neuron model, which captures the interactive dynamics between neuronal substructures and allows flexible configuration of compartmental numbers. This model is further formulated in a structured form that supports parallel training over long temporal sequences. In Chapter 5, we introduce a brain-inspired temporal structure encoding module to modulate neuronal dynamics, enabling spiking neurons to discriminate diverse temporal structures embedded in sensory inputs adaptively.

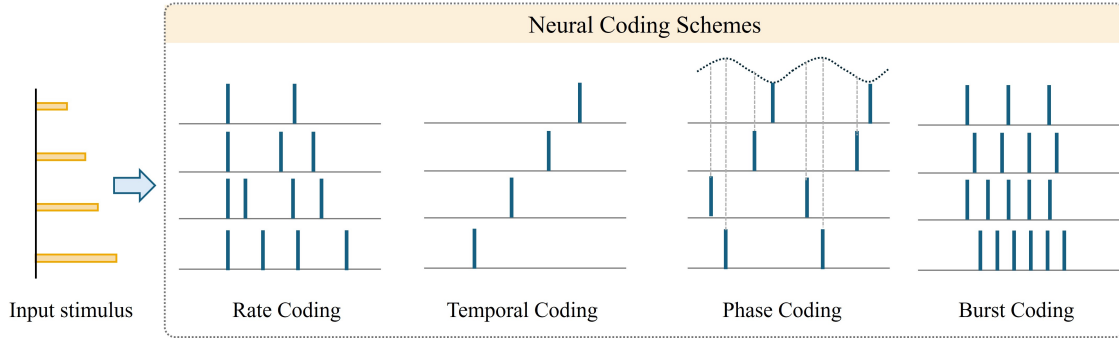


**Figure 2.3: Illustration of neural coding schemes observed in biological systems.** Adapted from [81].

## 2.3 Neural Coding Schemes

Beyond the neuron’s intrinsic dynamics and structures, another critical dimension in spiking neuron design lies in the spike-based neural coding schemes. Much like how human languages convey meaning through combinations of sounds and syntax, biological neurons communicate via spike trains characterized by distinct spatiotemporal patterns. These coding schemes define how sensorimotor signals are encoded into and decoded from spike trains, fundamentally shaping the expressiveness, robustness, and computational efficiency of spiking neurons. With the advancement of large-scale neural recording technologies, it is now well established that biological systems employ a rich diversity of coding schemes, which often coexist within the same network, highlighting the remarkable adaptability of biological neural systems. To better understand and replicate this diversity, as illustrated in Figure 2.3, recent studies have categorized neural coding strategies into several typical types, which are often observed with particular sensorimotor circuits and aligned with their characteristic task demands:

**Rate Coding.** The earliest research on neural coding can be traced back to 1926, when Adrian and Zotterman demonstrated that sensory nerves exhibit a firing rate that correlates with the weight applied to a muscle [82]. Since then, rate coding,

where information is encoded in the number of spikes within a time window, has become the most widely accepted scheme in both neuroscience and SNN research. Supporting evidence includes studies in the visual system and in motor control [83]. However, rate coding has inherent limitations. Accurate rate estimation often requires temporal averaging across long time windows or repeated trials, which contradicts the rapid decision-making observed in biological organisms. For instance, flies can react to new stimuli within 30–40 ms [84], and human object recognition can be completed within 150 ms [85], suggesting the need for coding strategies that rely on low-latency responses.

**Temporal Coding.** To address these limitations, temporal coding has been proposed as a complementary approach. Here, the precise spike timings are crucial information carriers. A representative strategy is the time-to-first-spike (TTFS) coding, in which the timing of the first spike after stimulus onset carries the critical information. The earlier the spike occurs, the stronger or more salient the input is. TTFS coding has been observed across various sensory systems, including the retina, auditory cortex, and gustatory pathways [86, 87, 88], enabling fast and sparse signal transmission.

**Phase Coding.** Biological neural systems exhibit rich neural oscillations spanning multiple frequency bands, from slow theta oscillations (4–8 Hz) to fast gamma oscillations (30–100 Hz) [89]. These oscillatory activities provide internal timing references that guide the spike-based information representation. Phase coding leverages this by encoding information into the phase of a spike within the ongoing oscillation cycle. When the same stimulus is presented repeatedly, spikes tend to occur at consistent phases relative to the oscillatory waveform, forming phase-locked firing patterns. Such phase coding schemes have been widely observed in the hippocampus [90], visual cortex [91], and olfactory bulb [92], where groups of neurons preserve a stable relative firing order. This temporal coordination not only supports sparse and reliable information transmission but also plays a crucial role in perception, navigation, and memory functionalities.

**Burst Coding.** Another biologically significant coding scheme is burst coding, where information is encoded in a brief sequence of spikes called a “burst”, characterized by short inter-spike intervals (ISIs). This coding scheme is prominent in thalamic, auditory, and hippocampal systems [93, 94], where the number of spikes, burst duration, or burst frequency can carry information about stimulus strength, novelty, or task-specific features. Compared with other coding schemes, burst coding offers higher reliability and noise tolerance at a cost of higher spike density.

The diversity of neural coding strategies observed in biological systems has inspired various spike-based neural coding schemes in SNNs. These schemes not only determine how information is represented in spike trains, but also shape the latency, robustness, and energy efficiency of SNNs across different tasks. Rate coding is the most widely adopted scheme in SNNs, especially in ANN-to-SNN conversion methods, which approximate the rectified linear unit (ReLU) activation function in ANNs with the spike rate of spiking neurons over a temporal window [28, 38]. While this approach facilitates high information fidelity, it often involves high inference latency and energy consumption. To reduce spike density while preserving information fidelity, temporal coding schemes have been extensively explored in SNNs. In TTFS coding, the stimulus is encoded by the timing of the first spike, where earlier spikes indicate stronger activations [95, 96, 97]. TTFS enables rapid inference with fewer spikes, making it ideal for low-power SNN deployment. However, TTFS-based models face challenges in gradient-based training due to non-differentiable timing and sparse spike outputs, often leading to gradient vanishing and dead neuron problems [96]. Another line of work explores phase coding, where spike timing is aligned to an internal oscillatory rhythm to convey phase-based information. While primitive phase-coded SNNs can reduce spike density significantly [98], they often struggle to handle high-dimensional, rapidly changing sensory streams. Burst coding has also attracted increasing attention for its robustness and reliability. Unlike rate coding, which spreads spikes over time, burst coding transmits information in compact packets of spikes with short ISIs

[99]. This redundancy improves signal fidelity and robustness to noise, but typically comes at a higher computational cost.

While most existing SNNs are still grounded on uniform neural coding schemes to achieve suboptimal performance due to their inherent strengths and limitations, this thesis proposes a hybrid coding and learning framework in Chapter 3. This framework allows for the flexible assignment of diverse coding schemes and corresponding spiking neurons to different parts of the network, achieving enhanced performance in terms of accuracy, robustness, latency, and energy efficiency.

## 2.4 Learning Algorithms for SNNs

The human brain exhibits remarkable adaptability to accommodate changing environments, enabling us to learn from experience, form memories, and adjust behavior. This capacity for adaptation is rooted in neural plasticity, the ability of neurons to modify their synaptic strengths and connectivity patterns in response to both external tasks and internal objectives.

In the context of SNNs, learning is formulated as the adjustment of parameters, such as synaptic weights, delays, and neuronal hyperparameters, to optimize performance metrics on specific tasks. In the early development of SNNs, learning is primarily guided by biologically inspired local learning rules, such as Hebbian learning [100] and spike-timing-dependent plasticity (STDP) [101, 102], which adjusted synaptic weights based on the relative timing of pre- and postsynaptic spikes. While these rules offer insights into the plasticity observed in biological systems, they are difficult to scale to deeper networks for complex tasks due to their lack of global credit assignment and optimization efficiency. To overcome these challenges, it is essential to develop learning algorithms specifically tailored for SNN. However, due to the non-differentiability of the spike generation function  $H(\cdot)$ , conventional gradient-based

training methods are not directly applicable to SNNs. Specifically, the derivative of the spike output  $S[t]$  with respect to the membrane potential  $V[t]$  is given by:

$$\frac{\partial S[t]}{\partial V[t]} = \begin{cases} \infty, & V = \theta \\ 0, & V \neq \theta \end{cases}. \quad (2.7)$$

This discontinuity poses a challenge for gradient backpropagation, where the gradient is almost always zero except at a single point. This phenomenon, referred to as the dead neuron problem, can severely hinder convergence speed and stability [25]. To address this, a wide range of specialized training algorithms for SNNs have been proposed, spanning from ANN-SNN conversion methods to gradient-based direct training methods adapted for spiking dynamics. In the following, we briefly review these representative training techniques relevant to the scope of this thesis.

One prominent class of methods is ANN-to-SNN conversion, which leverages the weights of well-trained artificial neural networks to construct equivalent spiking architectures [27, 28, 38, 103, 104, 105, 106, 107, 108]. This approach can be viewed as a form of teacher-student learning, where a high-precision ANN transfers its knowledge to a low-precision SNN by aligning their activation states. Specifically, by establishing a linear mapping function between the activation values of ANN neurons and the firing rates of spiking neurons within a fixed time window, the trained weights and biases of the teacher ANN can be systematically transferred to the student SNN. This enables the SNN to closely approximate the ANN’s behavior and achieve near-lossless accuracy on various pattern recognition tasks. However, these conversion techniques are typically designed under strict assumptions. Specifically, they are restricted to the IF neuron model, a simplified case of the LIF model without membrane potential decay. Moreover, these methods typically require large simulation time windows to ensure accurate approximation. These constraints limit the biological plausibility of conversion techniques and make them difficult to deploy efficiently on edge computing devices, where computation latency is one of the major requirements.

An alternative and increasingly popular direction is direct gradient-based training, which treats each spiking neuron as a recurrent unit with temporal state evolution. Following this formulation, the conventional BPTT approach can be applied to optimize the SNN parameters [24, 25, 26, 33, 109, 110, 111]. To address the non-differentiability of the spike activation function, continuous surrogate gradient functions are introduced to replace the discontinuous derivative defined in Eq. (2.7) with approximations during the backward pass. A common example is the rectangular function, which is defined as:

$$g'[t] = \frac{1}{a} \text{sign}(|V[t] - \theta| < \frac{a}{2}), \quad (2.8)$$

where  $a$  denotes the width of the approximation window around the threshold  $\theta$ . In the limit, this function converges to the idealized derivative of the spiking function:

$$\lim_{a \rightarrow 0^+} g'[t] = \frac{\partial S[t]}{\partial V[t]}, \quad (2.9)$$

thereby enabling precise approximation of the true gradient direction and facilitating effective optimization of network parameters.

Building on this foundation, recent studies have extended direct training methods with more advanced techniques to improve trainability. These efforts include enhanced surrogate gradient functions [27, 30], normalization strategies tailored for spiking dynamics [31, 112], and specialized loss functions for stabilizing temporal error propagation [32, 113]. In parallel, extensive efforts have been made to incorporate architectural innovations from ANNs into SNNs, such as residual connections [33, 114], convolution layers [34, 104], recurrent units [35, 115], and attention mechanisms [36, 116], thereby allowing end-to-end feature learning in deep SNNs. Despite these advancements, direct training methods still face several practical challenges. First, the mismatch between the true gradient and its surrogate approximation tends to accumulate across layers and timesteps, leading to performance degradation in deep network structures and small time windows. Second, the sequential nature of gradient computation in BPTT across temporal dimensions severely hinders effective

parallelization, resulting in prohibitively long training times for extended sequences and limiting the scalability of SNNs in tasks demanding long-range temporal modeling.

In this thesis, the gradient-based approaches are adopted to enable learning of spiking neuron models. In Chapter 3, we introduce a novel layer-wise learning method to mitigate the approximation mismatch, enabling effective training of SNNs with hybrid coding schemes. In Chapter 4, we further improve the training efficiency of BPTT by designing a novel spiking neuron model that enables parallel computation across time.

## 2.5 Neuromorphic Benchmarks

Benchmarking plays a vital role in assessing the current landscape of a research field and guiding its future directions. To evaluate the effectiveness of advanced spiking neuron models in practical pattern recognition scenarios, standardized neuromorphic benchmarks have become increasingly essential. These benchmarks should move beyond conventional tasks such as static image classification, and instead focus on real-world application scenarios that capture the unique advantages of NC. In addition, evaluation metrics should extend beyond standard accuracy scores to reflect the intrinsic characteristics of neuromorphic systems, including throughput, energy efficiency, latency, and robustness under deployment constraints. In recent years, a growing number of benchmarks, either widely used by conventional ANNs or neuromorphic-specific, have emerged to support the development of biologically inspired and energy-efficient intelligence. In the following, we review representative benchmarks relevant to this thesis. These benchmarks can be broadly categorized into three categories.

**Static Image Classification Benchmarks.** These benchmarks are originally de-

signed for ANNs and later adopted in early SNN studies to verify training stability and convergence behavior. Due to the inherent temporal nature of SNNs, these static datasets are typically converted into pseudo-temporal sequences before being fed into the network. A common strategy involves replicating each image along the time axis [62], enabling SNNs to process static inputs in a temporally-resolved manner and achieve performance comparable to their ANN counterparts. Representative datasets in this category include MNIST [117], a simple 10-class handwritten digit classification task; CIFAR-10 and CIFAR-100 [118], which consist of  $32 \times 32$  natural images across 10 and 100 classes, respectively; ImageNet [119], a large-scale classification dataset with over 1,000 categories, widely regarded as one of the most challenging benchmarks in neuromorphic visual recognition; and TinyImageNet [120], a mid-scale variant derived from ImageNet, contains 200 object categories with reduced image resolution to evaluate model generalization under constrained computational resources. These datasets primarily served as foundational tasks for validating SNNs' performance across varying levels of classification difficulty.

**Event-based Vision Benchmarks.** The second category encompasses event-based vision benchmarks generated using neuromorphic sensors, particularly dynamic vision sensors (DVS). These datasets operate in an event-driven and asynchronous manner with high temporal resolution, naturally align with the processing paradigm of SNNs on real-world dynamic vision tasks. These benchmarks can be further divided into two subtypes based on their data acquisition methodology. The first subtype involves converting conventional frame-based visual datasets into event streams by simulating camera motions, typically via artificial saccadic motions. This approach enables the reuse of well-established image datasets in neuromorphic settings. Representative examples include N-MNIST [121], CIFAR10-DVS [122], and N-Caltech101 [123], which are neuromorphic counterparts of MNIST [117], CIFAR-10 [118], and Caltech101 [124] datasets, respectively. The second subtype consists of datasets recorded directly using DVS cameras, capturing real-world dynamic scenes with native event streams. Repre-

sentative examples include DVS-Gesture [125], containing 11 gesture classes recorded under varying lighting conditions and across different subjects; DVS128-Gait-Day and DVS128-Gait-Night [126], focusing on gait identification under day and night light conditions, respectively; DVS-SLR [127], an action recognition dataset designed to evaluate the spatiotemporal modeling capabilities of SNNs; and HARDVS [128], a high-resolution event-based dataset comprising 300 challenging classes captured in complex real-world environments.

**Audio Classification Benchmarks.** This category offers testbeds for evaluating the ability of SNNs to model spatiotemporal dynamics, particularly short-term temporal dependencies common in speech signals. The audio classification datasets can be broadly divided into two subtypes. The first subtype includes conventional audio datasets such as Google Speech Commands (GSC) [129], which consists of common command keywords used in real-world applications, and TIMIT [130], a dataset designed for phoneme recognition, which provides high-quality speech signals with a wide variety of speakers and acoustic conditions. In these datasets, raw audio waveforms are first preprocessed using frequency-domain analysis methods, such as spectrogram or Mel-Frequency Cepstral Coefficients (MFCC) encoders, before being fed into the SNN. A second category involves event-based neuromorphic audio datasets, in which raw audio is converted into spike trains through biologically inspired auditory encoding techniques. Notable datasets include the Spiking Heidelberg Digits (SHD) and Spiking Speech Commands (SSC) [131], which contain spoken digits and command words in an event-based form. These datasets provide structured, event-driven input formats that are well-suited for evaluating low-latency, energy-efficient audio processing neuromorphic systems.

Despite these benchmarks having played a pivotal role in driving NC research over the past decade [62], many of these benchmarks lack rich temporal dynamics, failing to reflect the complex, multiscale temporal structures encountered in real-world pattern recognition tasks. As a result, SNN models evaluated solely on these bench-

marks may yield misleading conclusions regarding their actual temporal modeling capacity. Moreover, their application scenarios are largely limited to computer vision and speech tasks, overlooking many edge applications where the advantages of NC are most pronounced. Furthermore, most existing benchmarks primarily emphasize classification accuracy, while omitting broader system-level considerations that are critical for the practical deployment of neuromorphic models. In this thesis, we introduce a novel neuromorphic benchmark suite in Chapter 6, which fills these gaps by offering a suite of real-world temporal processing tasks that reflect the broad application spectrum of NC, along with standardized metrics that jointly assess accuracy, training speed, memory usage, and energy efficiency. This framework enables more rigorous and holistic evaluation of spiking neuron models, thereby facilitating their advancement toward real-world applications.

## Chapter 3

# A Hybrid Neural Coding Approach for Pattern Recognition with Spiking Neural Networks

This chapter advances spiking neuron models with enriched information representation strategies by introducing a hybrid neural coding and learning framework, which enhances the fidelity, efficiency, robustness, and adaptability of neural representations in spiking neurons. In contrast to conventional SNNs built upon homogeneous neuron models and uniform coding schemes, the proposed framework integrates a variety of neural coding schemes observed in neuroscience. It further enables flexible assignment of coding schemes to different network layers to accommodate task requirements. Moreover, it introduces novel layer-wise learning methods tailored for the effective training of hybrid coding SNNs. Experiments on representative pattern recognition tasks, including image classification and sound localization, demonstrate that the proposed framework not only achieves performance comparable to SOTA ANNs but also substantially reduces inference latency and energy consumption, while enhancing robustness against noise. These results provide practical insights into hy-

brid neuron model design and lay the foundation for constructing more accurate, robust, and adaptive neuromorphic pattern recognition systems.

## 3.1 Introduction

With rich neuronal dynamics and sparse spiking activities, SNNs are designed to mimic the remarkable information processing mechanisms of biological neural systems *in silico*. Recent advancements in SNN training algorithms [24, 25, 132] and model architecture designs [31, 33] have enabled a wide range of applications, including image classification [30, 116], speech processing [39, 133, 134], and robotic control [49, 50]. However, it remains an open question how to encode sensory signals into spike trains and to process them within SNNs to achieve optimal task performance. The SOTA SNNs used to solve pattern recognition tasks are primarily built on homogeneous spiking neuron models that share a uniform neural coding scheme. They further rely on extended learning with large-scale datasets to reach high classification accuracies. This homogeneous coding approach encounters challenges in effectively achieving multiple objectives that a system is expected to fulfill in practice, encompassing accuracy, efficiency, response latency, and robustness. For example, robotic platforms interact with environments via multimodal sensory signals such as vision, touch, and sound, each with distinct characteristics and processing demands. Vision involves high-dimensional spatial patterns that demand high fidelity, touch generates sparse but rapidly changing signals requiring low-latency responses for real-time interaction, while sound conveys fine-grained temporal structures necessitating precise temporal processing. Such heterogeneity prevents a single coding scheme from accommodating all modalities effectively.

The findings in biological neural systems prompt us to look into diversified neural coding schemes in neural computation. Over the last few decades, growing biological evidence indicates that perception and cognition are supported by diversified neu-

ral coding schemes, which are believed to play an essential role in signal processing, learning, and memory in neural systems [52]. The earliest research on neural coding can be traced back to 1926 when Adrian and Zotterman first revealed that sensory nerves innervating the muscle discharge spikes at a rate proportional to the weights hung from the muscle [82]. Since then, substantial evidence has been accumulated from different neural systems, leading to the conclusion that the firing rate of neurons is the key information carrier. However, the rate coding failed to account for many important features of neural computation. To address this limitation, temporal coding schemes have been proposed as complementary approaches to explain the limitations of rate coding. For example, in the human visual system, object recognition can be accomplished within 150 ms since the stimulus onset [85], during which signals need to travel across multiple interconnected stages. The rank-order coding has been proposed to explain this rapid information processing. It suggests the sensory information is represented in the chronological order of spikes, and more important features will be conveyed earlier [135]. Additionally, the spiking activities are found to be phase-correlated with the ongoing neural oscillations during memory formation in the hippocampus [90]. Moreover, as observed in the thalamus, hippocampus, and auditory systems, neurons are capable of firing a burst of spikes with short ISIs. This rapid information transmission mechanism has led to the development of burst coding [93].

The large body of research on neural coding has provided a wealth of inspiration for SNN designs. Nevertheless, most of the existing SNNs are only grounded on a single type of these neural coding schemes and corresponding spiking neuron models, overlooking the critical interplay between them. Consequently, this design methodology typically leads to suboptimal performance on the targeted task. For instance, direct coding uses analog values at the input or output layer, maintaining a high resolution of information but is the least biologically plausible and energy-efficient scheme. The majority of research on ANN-to-SNN conversion adopts rate coding, where the firing

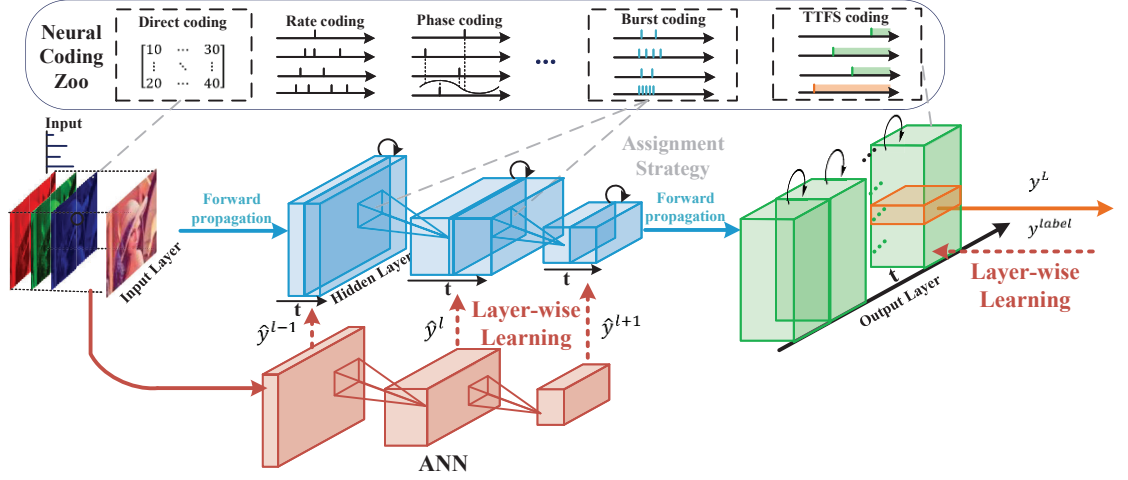


**Figure 3.1: Illustration of the proposed hybrid neural coding and learning framework in solving the image classification task.** The proposed framework consists of three components: Neural Coding Zoo (upper box), Assignment Strategy (gray dotted line), and Layer-wise Learning Method (red dotted line). First of all, the neural coding zoo is a comprehensive collection of the most representative neural coding schemes discovered in neuroscience. According to the specific task requirements of image classification, the assignment strategy has been further designed to assign heterogeneous neural coding schemes from the neural coding zoo to different network layers: direct coding for high-fidelity input representation, burst coding for reliable and rapid hidden feature representation, and TTFS coding to ensure fast and efficient output decision-making. Finally, the layer-wise learning methods are introduced to independently train the hidden and output layers, so as to achieve the desired coding schemes. Here,  $\mathbf{y}^L$  represents the outputs of the hybrid coding SNN,  $\mathbf{y}^{label}$  denotes the one-hot labels,  $\hat{\mathbf{y}}^l$  signifies the activation values of neurons at hidden layer  $l$  in the teacher ANN.

rate of IF neurons is used to approximate the output of the ReLU activation function [28, 38, 104, 108, 136]. While this approach achieves high accuracy, it typically requires long time windows, thereby introducing considerable latency during real-time inference. As a more reliable coding scheme, burst coding encodes information into ISIs and allows a burst of spikes to be generated within a short time interval for robust information transmission [99]. Despite its superior performance, burst coding often incurs a substantially higher spike density, resulting in increased computational cost. Some recent studies are looking at more efficient coding schemes. Phase coding,

for example, embeds information in the phase of spikes relative to ongoing neural oscillations, which remarkably reduces the spike density [98]. However, primitive phase coding struggles to represent a highly dynamic information stream. TTFS coding, on the other hand, encodes information via the first spike time [95, 96, 97]. However, these TTFS-based SNNs face gradient instability and dead neuron problems during training [96], which restrict their scalability.

In this chapter, we argue that SNNs should be holistically designed by incorporating various neural coding schemes to achieve optimal performance. To this end, we propose a hybrid neural coding and learning framework. As illustrated in Figure 3.1, the proposed framework incorporates a neural coding zoo formed by a collection of spiking neurons with diverse bio-inspired neural coding schemes. According to the assignment strategy and task requirements, these coding schemes are allocated to each network layer, thereby providing the desired information representational basis. Moreover, we introduce a layer-wise learning method to ensure the desired neural representation can be effectively achieved at each network layer. Consequently, the hybrid coding SNNs can take advantage of diversified coding schemes to achieve optimal task performance. We verify the effectiveness of the proposed framework on both image and audio perception tasks, and the results demonstrate the desired attributes of high accuracy and robustness, as well as low latency and energy consumption. The main contributions of this chapter are summarized as follows:

- This chapter presents a novel hybrid neural coding and learning framework, which advances spiking neuron models with enriched information representation strategies, allowing diverse coding schemes to be flexibly assigned to meet varying task demands and environmental conditions.
- This chapter proposes a layer-wise learning method to efficiently train the desired neural coding schemes with high levels of efficacy.
- This chapter showcases the superiority and versatility of the proposed framework

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

---

in performing image classification and sound localization tasks.

- Our research yields valuable insights into hybrid neuron model designs, which serve as practical guidelines for solving real-world pattern recognition tasks with hybrid coding SNNs.

The remainder of this chapter is organized as follows: in Section 3.2, we present the proposed hybrid neural coding and learning framework. We first introduce the diverse coding schemes included in the neural coding zoo and the assignment strategy designed to allocate different coding schemes to distinct network layers. We then illustrate how hybrid coding is implemented for image classification and sound localization tasks. Finally, a layer-wise learning method is introduced to enable effective training of the proposed hybrid coding approaches. Section 3.3 validates the effectiveness of the proposed framework through extensive experiments on image classification and sound localization tasks, and further conducts ablation studies to analyze the impact of different coding combinations. Finally, Section 3.4 summarizes the main findings of this chapter.

## 3.2 A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

In this section, we present our proposed hybrid neural coding and learning framework that provides a holistic way to design hybrid coding SNNs. As illustrated in Figure 3.1, our framework consists of three key components: neural coding zoo, hybrid coding assignment strategy, and layer-wise learning method. In the following, we begin with an overview of SNNs and their coding schemes. Building upon this, we then describe the collection of brain-inspired neural coding schemes that jointly form our neural coding zoo. We also elaborate on the strategy employed to assign these coding

schemes to different layers of the network. Furthermore, we take image classification and sound localization tasks to illustrate how a hybrid coding assignment strategy could be designed in practice to fulfill distinct task objectives. Finally, we present the proposed layer-wise learning method.

### 3.2.1 Neural Coding Zoo

Unlike traditional artificial neurons, spiking neurons are designed to reproduce the complex neuronal dynamics of biological neurons. Notably, spiking neurons use spike trains as information carriers to transmit information to each other. The method of how information is represented by a series of spike trains, a concept referred to as a neural coding scheme, significantly influences its information processing capability. In general, the frequency, precise spike timing, and synchrony patterns within the spike trains could all be utilized to carry information. An appropriate assignment of these neural coding schemes to an SNN could significantly enhance its information processing efficiency, efficacy, and robustness.

Inspired by the rich biological neural representations uncovered in neuroscience, a variety of neural coding schemes, each with its unique strengths and limitations, have been explored for SNNs. In our proposed framework, the neural coding zoo is a comprehensive collection of the most representative neural coding schemes, which provides a wealth of candidate options for designing hybrid coding SNNs. It includes direct coding, rate coding, phase coding, burst coding, and TTFS coding, facilitating the construction of versatile SNNs that can meet diversified task requirements and adapt to various environmental conditions. In the following, we provide a detailed summary of each candidate neural coding scheme, with their merits and weaknesses depicted in Figure 3.2. It is worth noting that our neural coding zoo can effortlessly integrate any newly discovered or developed neural coding schemes.

- **Direct coding** has been widely used in input and output layers of SNNs, which di-

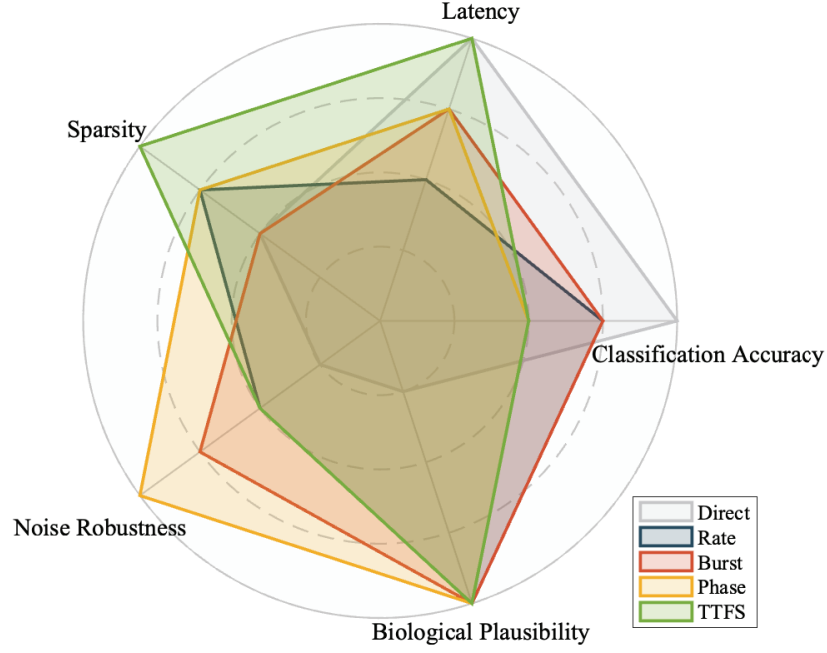


**Figure 3.2: Comparison of computational performance and biological plausibility among diverse candidate neural coding schemes.**

rectly employs analog values to encode input stimuli or to decode outputs [39, 132]. Given its analog nature, it is the least biologically plausible and energy-efficient coding scheme, as it abandons the binary spike-based information representation. Nevertheless, direct coding could avoid sampling errors and delays inherent in other coding schemes, thus ensuring a high fidelity in information representation.

- **Rate coding** employs the mean firing rate within a given time window to signify the signal value. Due to the similarity between ReLU activation values and firing rates of spiking neurons, rate coding has been widely adopted in ANN-to-SNN conversion methods [28, 38, 104, 108, 136] to construct SNNs that can well approximate the feature representation of ANNs. Despite its biological plausibility and superior information representation capability, rate coding is often criticized for high latency, poor energy efficiency, and ineffectiveness in representing highly dynamic signals.
- **Phase coding** represents information through the phase of spiking activities rel-

ative to underlying neural oscillations [137]. It is proven to be more efficient in information transmission than rate coding and offers higher noise robustness than TTFS coding [138]. However, the representation capacity of phase coding is constrained by the frequency of underlying neural oscillations, which leads to reduced performance when tasked with representing rapidly changing stimuli [98, 139].

- **Burst coding** makes effective use of the ISI to represent information [140] and is highly efficient in information transmission [81]. However, keeping track of the ISI introduces a high computational complexity. To address this issue, recent works propose using a neuron model that can emit multiple spikes within a single timestep to represent graded ISIs [99, 141], resulting in a simple yet robust neural coding scheme.
- **TTFS coding** encodes information based on the timing of the first spike, where earlier spikes represent more significant information. This coding scheme leads to the favorable properties of fast inference, low spike density, and low energy consumption [142, 143]. However, TTFS coding suffers from the notorious dead neuron problem, which can affect the training stability. Furthermore, spike timing is highly sensitive to neuronal noise, a common occurrence in neural systems and analog computing substrates.

### 3.2.2 Hybrid Coding Assignment Strategy

As discussed earlier, each homogeneous neural coding scheme possesses its unique merits and weaknesses, and if applied alone, it will significantly impact the overall performance of SNNs. Therefore, our objective is to design hybrid coding SNNs that can leverage the benefits of these homogeneous neural coding schemes to achieve optimal task performance. This can be achieved by selecting appropriate neural coding schemes from our neural coding zoo and assigning them to different parts of the network according to varying task requirements and environmental conditions. However,

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

---

this will lead to a complex combinatorial optimization problem that is intractable for large-scale networks. To reduce the design effort and accelerate training, we organize the network layers into three blocks, each of which adopts a homogeneous neural coding scheme throughout its entirety. Specifically, we focus on three network blocks: Input Layer, Hidden Layers, and Output Layer. In the following, we describe the critical factors when assigning neural coding schemes to these three network blocks.

- **Input Layer** plays a significant role as the interface between an SNN and its external environment. It is crucial for this layer to provide an efficient, real-time, and high-fidelity representation of the input stimuli. To achieve these objectives, the input coding scheme must be tailored to the nature of the input signal and the prevailing environmental conditions, ensuring a fast response while minimizing information loss during the sampling and encoding process.
- **Hidden Layers** play a pivotal role in extracting rich and hierarchical feature representations from the input stimuli. Consequently, it is imperative that the chosen coding scheme not only facilitates efficient feature extraction but also balances between computational cost and robustness to neuronal noise.
- **Output Layer** functions as the output interface for decision generation, thereby underscoring the need for a neural coding scheme that can simultaneously facilitate accurate and fast decision-making.

This assignment strategy serves as a general guideline on how to construct an optimal hybrid coding SNN that can meet task-specific objectives and environmental conditions.

#### 3.2.3 Hybrid Coding Design for Image Classification

In this subsection, grounded on the neural coding zoo and assignment strategy in-

troduced earlier, we will illustrate how a hybrid coding SNN can be practically constructed to tackle the image classification task.

### Direct Coding for High-Fidelity Input Representation

In image classification tasks, a high-fidelity representation of input images is typically required. However, the use of rate coding may require a large sampling time window and high firing rates, leading to slow and inefficient runtime. On the other hand, temporal coding schemes (e.g., TTFS coding) may encounter issues like dead neurons and gradient instability, adversely affecting the ease of training [96]. To address these challenges, we adopt a direct coding scheme that treats the intensity value of image pixels as the input current and directly injects it into the neurons in the first hidden layer [39, 132]. This approach bypasses the analog-to-spike conversion process, eliminating the information loss during rate-based sampling as well as the quantization errors during temporal encoding.

### Burst Coding for Reliable and Rapid Hidden Feature Representation

In contrast to rate coding, which restricts neurons to emit at most one spike at a time, bursting neurons can generate up to  $\Gamma$  spikes. This characteristic facilitates faster transmission of important information with greater robustness. Therefore, we employ burst coding to hidden layers to ensure rapid and efficient transfer of spike-based feature representations to subsequent layers. Specifically,  $\Gamma$  is set to 5 considering that neurons rarely emit a spike burst with more than five spikes in a biological discovery [144]. We utilize the bursting neuron model introduced in [141], in which the membrane potential is charged as:

$$\mathbf{V}^l[t] = \mathbf{V}^l[t-1] + \mathbf{I}^l[t], \quad (3.1)$$

where  $\mathbf{I}^l[t]$  denotes the input current induced from the input spikes, which is defined as  $\mathbf{I}^l[t] = \mathcal{W}^l \mathbf{S}^{l-1}[t]$ .  $\mathbf{S}^{l-1}[t]$  is the input spikes from layer  $l-1$ ,  $\mathcal{W}^l$  represents the

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

weight matrix between layer  $l$  and  $l - 1$ . Once the membrane potential  $\mathbf{V}^l$  exceeds the firing threshold  $\theta^l$ , an output spike burst  $\mathbf{S}^l$  will be generated, and the membrane potential  $\mathbf{V}^l$  will be reset using the reset-by-subtraction scheme [145] as:

$$\begin{aligned}\mathbf{S}^l[t] &= \min \left( \max \left( \lfloor \frac{\mathbf{V}^l[t]}{\theta^l} \rfloor, 0 \right), \Gamma \right), \\ \mathbf{V}^l[t] &= \mathbf{V}^l[t] - \mathbf{S}^l[t] \cdot \theta^l,\end{aligned}\tag{3.2}$$

where the  $\mathbf{S}^l[t]$  signifies the bursting spike count in the timestep  $t$ , whose value is taken from range  $[0, \Gamma]$ ,

#### TTFS Coding for Fast and Efficient Output Decision-making

Subsequent to the image feature extraction process in the hidden layers, the output layer is responsible for decision-making. Ideally, the classification decision should be made as soon as sufficient evidence has been accumulated at the output layer  $L$ . To achieve this, we adopt TTFS coding that reaches the final decision once the first output spike has been detected, without waiting for all simulation timesteps to finish. This design thus leads to a significant reduction in inference time while effectively circumventing the gradient instability issue commonly associated with temporal coding, as it is confined to a single-layer implementation. Specifically, we adopt the double exponential LIF spiking neuron model. It offers richer temporal dynamics than the bursting neuron model used in hidden layers and is defined as [146] :

$$\begin{aligned}\mathbf{V}^L[t] &= \mathcal{W}^L \mathbf{K}^L[t], \\ \mathbf{S}^L[t] &= H(\mathbf{V}^L[t] - \theta^L),\end{aligned}\tag{3.3}$$

where  $H(\cdot)$  denotes the Heaviside step function. Once the membrane potential  $V_i^L[t]$  exceeds the firing threshold  $\theta^L$ , the  $i^{th}$  neuron emits a spike at timestep  $t$ .  $\mathbf{K}^L[t]$  is the post-synaptic potential (PSP) kernel induced from input spikes as [146]:

$$\begin{cases} \mathbf{K}^L[t] = K_0 \cdot (\mathbf{m}^L[t] - \mathbf{I}^L[t]), \\ \mathbf{m}^L[t] = \exp(-\frac{1}{\tau_m}) \cdot (\mathbf{m}^L[t-1] + \mathbf{S}^{L-1}[t]), \\ \mathbf{I}^L[t] = \exp(-\frac{1}{\tau_s}) \cdot (\mathbf{I}^L[t-1] + \mathbf{S}^{L-1}[t]), \end{cases}\tag{3.4}$$

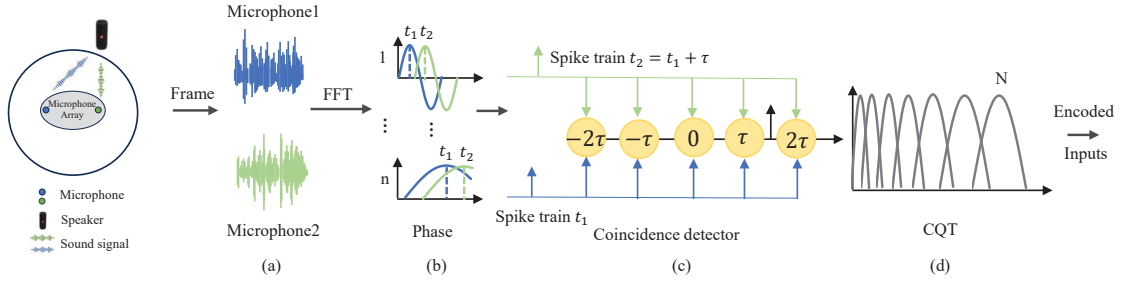


**Figure 3.3: Illustration of the phase encoding front-end used for the sound localization task, where two microphones are selected for demonstration. (a)** The raw audio is decomposed into  $n$  single-tone sinusoidal signals by fast Fourier transform (FFT). **(b)** The first peak time of each sinusoidal signal is encoded into a precise-timing spike, which represents the arrival time of each sound. **(c)**  $N_\tau$  coincidence detection neurons are employed to detect the phase difference between two microphones. If a coincidence detection neuron detects its predefined phase difference, it emits a spike. **(d)** To reduce the computational cost of post-processing, the encoded spike trains of the  $n$  single tones are further grouped into  $N_c$  channels via CQT.

where  $K_0$  is a normalization constant that aligns the maximum value of  $\mathbf{K}^L$  with  $\theta^L$ ,  $\tau_m$  and  $\tau_s$  are decay constants of two synaptic components  $\mathbf{m}^L$  and  $\mathbf{I}^L$ , respectively.

The classification result  $\mathbf{y}^L$  is determined by the output neuron that generates the first spike among all output neurons, with the trigger time denoted as  $t_f$ . In edge cases where multiple neurons fire simultaneously, the decision is made according to the neuron with the highest membrane potential:

$$\mathbf{y}^L = \max_i (V_i^L[t_f]). \quad (3.5)$$

where  $V_i^L[t_f]$  denotes the potential of neuron  $i$  at the first spike time  $t_f$ . Furthermore, if no neuron fires before the end of the simulation, we treat the last timestep as the decision time and determine the classification result according to Eq. (3.5).

### 3.2.4 Hybrid Coding Design for Sound Localization

In this subsection, we further employ the proposed hybrid neural coding and learning framework to solve the sound localization task, demonstrating its generalizability

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

---

across different signal modalities and task requirements.

#### **Phase Coding for High-fidelity Input Representation**

Mammals possess a remarkable capability to figure out the origin of the sound, as they can discern the time difference of arrival (TDOA) between two ears [147]. Specifically, sound from the same source propagates and reaches two ears with different time delays and energy attenuation, which serve as key cues for sound localization. To detect these discrepancies, each cochlea first employs phase coding to encode the disparate sound into phase-locked spike trains. Subsequently, the detection neurons in the medial superior olive (MSO) can discern the phase differences between these spike trains, thus enabling sound localization [148].

Drawing inspiration from the temporal characteristics of sound signals and the biological information representation mechanism, phase coding emerges as an ideal choice for extracting and encoding localization cues. Therefore, we adopt the multi-tone phase coding (MTPC) [44] as the input coding scheme within our framework to ensure a high-fidelity representation of the input. As depicted in Figure 3.3, the received audio is first decomposed into several single-tone sinusoidal signals. Then, a spike is generated at the first peak time of each sinusoidal signal, encoding the arrival time of the sound. The coincidence detection neurons then detect the phase difference between two microphones, generating an output spike if a specific time delay is observed. Finally, the output spikes generated from sub-bands are aggregated using constant Q transform (CQT) to compress the input dimensions.

#### **Burst Coding for Robust Hidden Feature Representation**

In audio signal processing, rapid information transmission and robust information representation are two desirable attributes when selecting a coding scheme for hidden layers. Low latency can significantly enhance the overall robustness of the system by

allowing it to respond swiftly to changes in the input signals. Concurrently, robust information representation can effectively overcome the ubiquitous neuronal noise. To this end, we select burst coding for hidden layers, as it demonstrates superior performance in these two aspects, as shown in Figure 3.2.

### **TTFS Coding for Fast and Efficient Output Decision-Making**

High accuracy, swift response to moving sound sources, and superior energy efficiency are critical attributes for any auditory interface. These factors motivate our choice of TTFS coding for the output layer of the localization network. In Section 3.3.2, we will illustrate how TTFS coding effectively reduces inference time without compromising localization accuracy. Furthermore, in Section 3.3.5, we will compare TTFS coding with other coding schemes to further validate its effectiveness in the context of sound localization.

#### **3.2.5 A Layer-wise Learning Method for Hybrid Coding**

Despite the flexibility that hybrid coding SNNs offer in assigning different coding schemes across various network stages, it presents its own set of challenges. Particularly, the use of different neural coding schemes renders the conventional end-to-end training approach incompatible. To resolve this challenge, we propose a novel layer-wise training method that is tailored to accomplish the designed hybrid coding SNN. In particular, the hidden and output layers are trained independently in a more manageable way.

##### **Hidden Layer**

For all hidden layers, a layer-wise Teacher-Student learning approach is adopted. Specifically, our method leverages a pre-trained ANN as a teacher, to supervise the

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

training of a student SNN. Considering the intermediate feature representations of the ANN as the targets, the hidden layers of our hybrid coding SNNs are trained to generate equivalent spike-count-based feature representations. For this purpose, the mean square error (MSE) between ANN and SNN features is utilized as the local loss function for each hidden layer, which is defined as:

$$\mathcal{L}^l(\hat{\mathbf{y}}^l, \mathbf{c}^l(T)) = \left\| \hat{\mathbf{y}}^l - r^l \cdot \frac{\mathbf{c}^l(T)}{T} \right\|_2^2, \quad (3.6)$$

where  $\hat{\mathbf{y}}^l$  denotes the activation of neurons in teacher ANN layer  $l$ .  $T$  is the time window length and  $\mathbf{c}^l(T) = \sum_{t=1}^T \mathbf{S}^l[t]$  is the total spike count.  $r^l$  is a layer-wise scaling factor whose value is fixed to 1. This ensures the firing rate of spiking neurons, denoted as  $\mathbf{c}^l(T)/T$ , can effectively align with the activation value of ANN neurons.

To eliminate the mismatch between training and testing, we train each hidden layer by taking into consideration the temporal dynamics of input spike trains rather than only considering the mean firing rate as in [108]. Specifically, the BPTT algorithm is employed to perform temporal credit assignments at each layer. The weight update can be derived as follows:

$$\Delta \mathcal{W}^l \propto \frac{\partial \mathcal{L}^l}{\partial \mathcal{W}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}^l}{\partial \mathbf{V}^l[t]} \frac{\partial \mathbf{V}^l[t]}{\partial \mathcal{W}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}^l}{\partial \mathbf{V}^l[t]} \mathbf{S}^{l-1}[t]. \quad (3.7)$$

For the ease of expression, we let  $\delta^l[t] \triangleq \frac{\partial \mathcal{L}^l}{\partial \mathbf{S}^l[t]}$  and yield

$$\frac{\partial \mathcal{L}^l}{\partial \mathbf{V}^l[t]} = \begin{cases} \delta^l[t+1] \frac{\partial \mathbf{S}^l[t+1]}{\partial \mathbf{V}^l[t+1]} + \delta^l[t] \frac{\partial \mathbf{S}^l[t]}{\partial \mathbf{V}^l[t]}, & \text{if } t < T, \\ \delta^l[T] \frac{\partial \mathbf{S}^l[T]}{\partial \mathbf{V}^l[T]}, & \text{if } t = T, \end{cases} \quad (3.8)$$

where

$$\delta^l[t] = \begin{cases} -\theta^l \delta^l[t+1] \frac{\partial \mathbf{S}^l[t+1]}{\partial \mathbf{V}^l[t+1]} + \delta^l[T], & \text{if } t < T, \\ -\frac{2}{T} (\hat{\mathbf{y}}^l - r^l (\frac{1}{T} \sum_{t=1}^T \mathbf{S}^l[t])), & \text{if } t = T. \end{cases} \quad (3.9)$$

To address the discontinuity in burst neurons as Eq. (3.2), we adopt the straight-through estimator (i.e.,  $d[x]/dx = 1$ ) [149] and result in  $\frac{\partial \mathbf{S}^l[t]}{\partial \mathbf{V}^l[t]} = 1$ . This local learning

methodology, henceforth referred to as Local Tandem Learning (LTL), broadens the scope of the original rate-coding-focused LTL rule [29] to accommodate additional neural coding schemes.

## Output Layer

To achieve fast and accurate decision-making, the output layer is trained to make correct classification decisions based on the first output spike, which is referred to as TTFS learning hereafter. It is accomplished by carefully balancing two learning objectives. On the one hand, the weights of the target neuron should be adjusted to fire as early as possible to reduce inference latency. On the other hand, all the other neurons should fire later than the target one to ensure decision accuracy. These two learning objectives are optimized together by adjusting the weights of the output layer according to the following two loss functions, while keeping all other hidden layer weights fixed.

To achieve the first objective, we select the cross-entropy (CE) loss as the first loss function, which advances the first spike time of the target neuron by forcing its membrane potential to increase during training as:

$$\mathcal{L}_1^L = -\ln \frac{\exp(V_{target}^L[t_f])}{\sum_{i=1}^n \exp(V_i^L[t_f])}, \quad (3.10)$$

where  $V_i^L[t_f]$  is the potential of the output neuron  $i$  at the first spike time  $t_f$ ,  $V_{target}^L[t_f]$  denotes that of the neuron corresponding to the ground-truth class,  $n$  is the total number of classes.

Given the fixed firing threshold and growing membrane potentials that result from minimizing  $\mathcal{L}_1$ , the decision time will be continuously advanced until every decision is made at the earliest possible timestep. This may, however, lead to an overfitting problem as the available input information is inadequately exploited. To address this issue, we introduce another loss, namely  $\mathcal{L}_2$ , to suspend incorrect decisions and allow

### 3.2. A Hybrid Neural Coding and Learning Framework for Spiking Neural Networks

---

more time to exploit input information. Specifically, long-term depression (LTD) will be triggered when any other neuron fires earlier than the target one as:

$$\mathcal{L}_2^L = \sum_{i \neq target}^n \gamma(t_f) \cdot (V_i^L[t_f] - \theta^L), \quad (3.11)$$

where  $\gamma(t_f)$  is a scaling factor that is defined as:

$$\gamma(t_f) = \max \left( 1 - \frac{acc(t_f)}{acc(T)}, 0 \right), \quad (3.12)$$

where  $acc(t_f)$  and  $acc(T)$  correspond to the inference accuracy at the first spike time and the last timestep, respectively, whose value is updated after each evaluation. For incorrect decisions made too early, their firing time will be delayed to later timesteps depending on whether the accuracy can be improved if the decisions are made later, that is  $\gamma(t_f) > 0$ .

The combination of the above two loss functions ensures those easy decisions can be made early while difficult decisions are postponed to later and more informative timesteps. The weight update of the output layer can be formulated as follows:

$$\Delta \mathcal{W}^L \propto \frac{\partial \mathcal{L}^L}{\partial \mathcal{W}^L} = \frac{\partial \mathcal{L}^L}{\partial \mathbf{V}^L[t_f]} \mathbf{K}^L[t_f], \quad (3.13)$$

where the total loss  $\mathcal{L}^L = \lambda_1 \mathcal{L}_1^L + \lambda_2 \mathcal{L}_2^L$  is tuned by two hyperparameters  $\lambda_1$  and  $\lambda_2$ . By moderating the ratio of  $\lambda_2/\lambda_1$ , we strike a balance between the classification accuracy and the inference latency. Depending on the specific task requirements, a large  $\lambda_2/\lambda_1$  favors accuracy, and a small value minimizes inference latency. Here, we employ the metric  $\Delta \text{Acc.} \times t_{inf}$  as a means to trade off between accuracy and inference latency.  $\Delta \text{Acc.}$  denotes the accuracy drop from ANN to SNN, and  $t_{inf}$ , namely the average inference time, is equivalent to the mean first spike time  $t_f$  across different samples.

### 3.3 Experimental Results

In this section, we validate the effectiveness of the proposed hybrid neural coding and learning framework on a wide range of pattern recognition tasks, including image classification and sound localization. We first compare the proposed hybrid coding approach to other SOTA SNN implementations on the CIFAR-10 [118], CIFAR-100 [118], and Tiny-ImageNet [120] datasets, in terms of two primary performance indicators of image classification tasks: classification accuracy and inference latency [150]. For all datasets, we utilize both VGG-16 [151] and ResNet-20 [152] network structures. We then validate the effectiveness and convergence speed of our proposed layer-wise learning method. We further demonstrate that the proposed approach can achieve SOTA classification accuracy while significantly reducing the computational cost. Subsequently, we assess real-world sound localization on the SLoClas dataset [153], analyzing localization precision, inference latency, energy consumption, and robustness to noise. Finally, several ablation studies are carried out to gain insight into the impact of different hybrid neural coding combinations on the overall performance. For simplicity, the direct, rate, burst, phase, and TTFS coding schemes are hereafter denoted as D, R, B, P, and T, respectively.

#### 3.3.1 Superior Image Classification Performance

In Table 3.1, we report our results and compare them against other SOTA SNN models. To ensure a fair comparison, we select competitive SNN benchmarks based on their conversion errors for network conversion methods, or classification accuracy for direct training methods. Overall, as the inference curves illustrated in Figure 3.4, the proposed hybrid coding approach attains competitive classification accuracy while significantly reducing inference time across all model architectures and datasets.

For the CIFAR-10 dataset, our VGG-16 model requires only 4.87 timesteps on average

### 3.3. Experimental Results

**Table 3.1: Comparison of classification accuracy and inference time on the CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets.  $\Delta\text{Acc.} = \text{SNN Acc.} - \text{ANN Acc.}$**

| Dataset       | Method                      | Architecture           | ANN Acc. (%) | SNN Acc. (%) | $\Delta$ Acc. (%) | Avg. Inference Time |
|---------------|-----------------------------|------------------------|--------------|--------------|-------------------|---------------------|
| CIFAR-10      | Burst Spikes [99]           | VGG-16                 | 91.41        | 91.41        | 0                 | 793                 |
|               | RMP [106]                   | VGG-16                 | 93.63        | 93.04        | -0.59             | 256                 |
|               | TSC [154]                   | VGG-16                 | 93.63        | 93.27        | -0.36             | 128                 |
|               | Calibration [108]           | VGG-16                 | 95.72        | 95.14        | -0.58             | 64                  |
|               | Burst+LIPooling [141]       | VGG-16                 | 95.74        | 95.58        | -0.16             | 32                  |
|               | QCFS [28]                   | VGG-16                 | 95.52        | 95.40        | -0.12             | 16                  |
|               | Diet-SNN [155]              | VGG-16                 | 93.72        | 93.44        | -0.28             | 10                  |
|               | <b>Ours (Hybrid Coding)</b> | <b>VGG-16</b>          | <b>95.84</b> | <b>95.66</b> | <b>-0.17</b>      | <b>4.87</b>         |
|               | RMP [106]                   | ResNet-20              | 91.47        | 91.36        | -0.11             | 2048                |
|               | TSC [154]                   | ResNet-20              | 91.47        | 91.42        | -0.05             | 2048                |
|               | Calibration [108]           | ResNet-20              | 95.46        | 94.78        | -0.68             | 32                  |
|               | Burst+LIPooling [141]       | ResNet-20              | 96.56        | 96.11        | -0.45             | 32                  |
|               | QCFS [28]                   | ResNet-20              | 91.77        | 91.62        | -0.15             | 16                  |
|               | Diet-SNN [155]              | ResNet-20              | 92.79        | 92.54        | -                 | 10                  |
|               | STBP-tdBN [31]              | ResNet-19*             | -            | 93.16        | -                 | 6                   |
|               | TET [32]                    | ResNet-19*             | -            | 94.50        | -                 | 6                   |
|               | <b>Ours (Hybrid Coding)</b> | <b>ResNet-20</b>       | <b>95.45</b> | <b>95.24</b> | <b>-0.21</b>      | <b>5.04</b>         |
| CIFAR-100     | Burst Spikes [99]           | VGG-16                 | 68.77        | 68.69        | -0.08             | 3000                |
|               | RMP [106]                   | VGG-16                 | 71.22        | 70.93        | -0.29             | 2048                |
|               | TSC [154]                   | VGG-16                 | 71.22        | 70.97        | -0.25             | 2048                |
|               | Calibration [108]           | VGG-16                 | 77.89        | 76.64        | -1.25             | 64                  |
|               | Burst+LIPooling [141]       | VGG-16                 | 78.49        | 78.26        | -0.23             | 64                  |
|               | QCFS [28]                   | VGG-16                 | 76.28        | 76.24        | -0.04             | 16                  |
|               | Diet-SNN [155]              | VGG-16                 | 71.82        | 69.67        | -2.15             | 5                   |
|               | <b>Ours (Hybrid Coding)</b> | <b>VGG-16</b>          | <b>77.15</b> | <b>75.75</b> | <b>-1.40</b>      | <b>6.88</b>         |
|               | RMP [106]                   | ResNet-20              | 68.72        | 67.82        | -0.90             | 2048                |
|               | TSC [154]                   | ResNet-20              | 68.72        | 68.18        | -0.54             | 2048                |
|               | Calibration [108]           | ResNet-20              | 77.16        | 76.32        | -0.84             | 32                  |
|               | Burst+LIPooling [141]       | ResNet-20              | 80.69        | 79.83        | -0.86             | 64                  |
|               | QCFS [28]                   | ResNet-20              | 69.94        | 69.82        | -0.12             | 32                  |
|               | Diet-SNN [155]              | ResNet-20              | 64.64        | 64.07        | -0.57             | 5                   |
|               | TET [32]                    | ResNet-19*             | -            | 74.72        | -                 | 6                   |
|               | <b>Ours (Hybrid Coding)</b> | <b>ResNet-20</b>       | <b>76.56</b> | <b>75.39</b> | <b>-1.17</b>      | <b>3.94</b>         |
| Tiny-ImageNet | Spike-thrift [156]          | VGG-16                 | 56.56        | 51.92        | -4.64             | 150                 |
|               | Calibration [108]           | VGG-16 <sup>+</sup>    | 60.95        | 53.96        | -6.99             | 32                  |
|               | Burst+LIPooling [141]       | VGG-16 <sup>+</sup>    | 61.80        | 55.56        | -6.24             | 64                  |
|               | QCFS [28]                   | VGG-16 <sup>+</sup>    | 54.91        | 54.92        | 0.01              | 16                  |
|               | <b>Ours (Hybrid Coding)</b> | <b>VGG-16</b>          | <b>58.75</b> | <b>56.95</b> | <b>-1.80</b>      | <b>7.11</b>         |
|               | Calibration [108]           | ResNet-20 <sup>+</sup> | 58.29        | 52.79        | -5.5              | 32                  |
|               | QCFS [28]                   | ResNet-20 <sup>+</sup> | 55.81        | 52.32        | -3.49             | 16                  |
|               | <b>Ours (Hybrid Coding)</b> | <b>ResNet-20</b>       | <b>58.34</b> | <b>56.50</b> | <b>-1.84</b>      | <b>6.54</b>         |

\* Only the results of ResNet-19 are reported in these works, which have comparable model capacity to other ResNet-20 models.

+ Our reproduced results using publicly available codes.

to reach 95.66% classification accuracy. Whereas the QCFS [28], considered the best among all ANN-to-SNN conversion methods, can only achieve a comparable accuracy

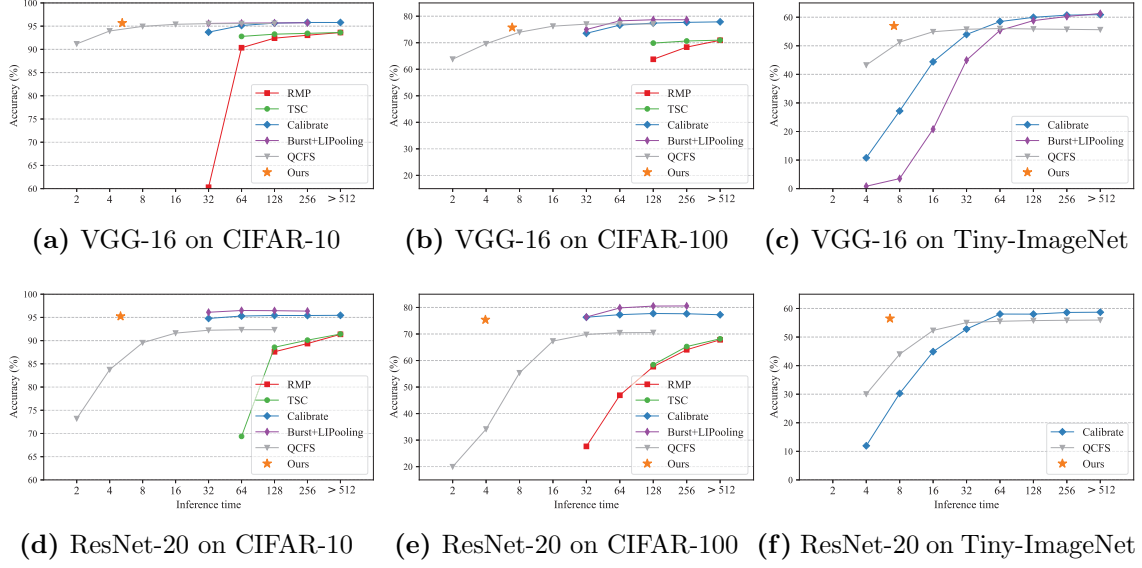


**Figure 3.4: Test accuracy as a function of inference time on the CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets with (a-c) VGG-16 and (d-f) ResNet-20 architectures.** Note that for the proposed hybrid coding approach, we only report the average inference time as the decision is made as soon as the first output spike is generated.

with 16 timesteps. Similar conclusions can be drawn from the converted ResNet-20 models. Furthermore, our ResNet-20 model outperforms other directly trained models, such as STBP-tdBN [31] and TET [32].

For the more challenging CIFAR-100 dataset, our model reaches a satisfactory accuracy of 75.75% and 75.39% with average inference timesteps of 6.88 and 3.94 for VGG-16 and ResNet-20, respectively. As presented in Figures 3.4(b, e), although the classification accuracy of our models is slightly worse than Calibration [108], Burst+LIPooling [141], and QCFS models, it is worth noting that their results are achieved with at least 16 timesteps. Consequently, our models are more favorable in terms of inference time and energy consumption. Additionally, we evaluate our method on a large-scale Tiny-ImageNet dataset. Due to a lack of benchmarks on this dataset, we reproduce the results for Calibration, Burst+LIPooling, and QCFS methods using their publicly available codes. As shown in Figures 3.4(c, f), it is

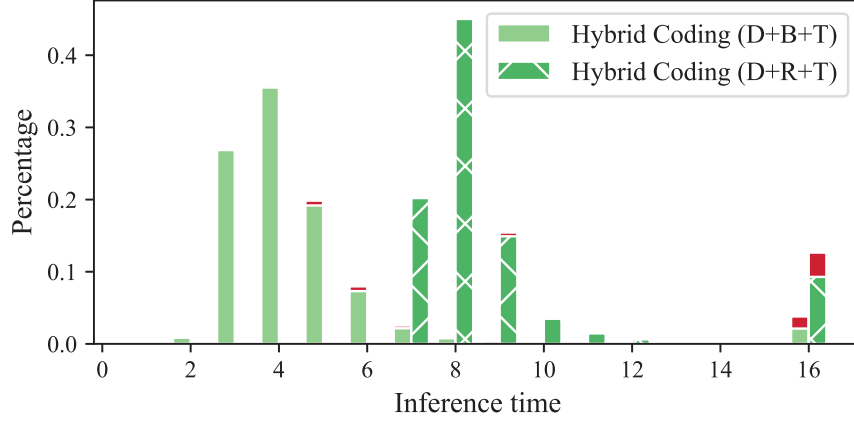


**Figure 3.5: Decision time distribution for the proposed hybrid coding models.** The decision times for each sample vary according to the TTFS coding mechanism. **Green Bar:** the proportion of correct decisions. **Red Bar:** the proportion of misclassified samples.

obvious that our proposed models are capable of achieving superior results with a shorter inference time.

To elucidate how early decision-making is accomplished with the proposed hybrid coding SNN, we present in Figure 3.5 the distribution of decision time across two of our hybrid coding models, D+B+T and D+R+T. While the other SOTA works listed in Table 3.1 employ direct coding in the output layer, resulting in a constant decision time of  $T$ . In contrast, our hybrid coding models exhibit varying decision times governed by the decision-making mechanism of TTFS coding. Furthermore, it is evident that the first spike time is densely distributed around 4 for D+B+T and 8 for D+R+T models, respectively. This result substantiates that burst coding facilitates faster information transmission across hidden layers, which forms the basis for the early decisions made by the TTFS classifier. Additionally, it is noteworthy that decisions made at earlier timesteps exhibit higher accuracy than those made later, as indicated by the red bars. This observation suggests early decisions typically correspond to simpler samples, and TTFS coding is capable of identifying when sufficient evidence has been accumulated for different samples. Compared with the direct decoding that makes all decisions at the final timestep, TTFS coding can significantly

reduce average inference time yet achieve high classification accuracy.

### 3.3.2 Effective and Rapid Training Convergence with Layer-wise Learning Method

To further explore the effectiveness and training convergence speed of the proposed layer-wise learning method, we conduct a series of studies in this subsection. All experiments are performed on the CIFAR-10 dataset using VGG-16 architecture.

Firstly, to assess the effectiveness of the LTL rule in producing the desired hidden layer representations as specified by the neural coding scheme, we systematically compare the performance of the Calibration model [108], the LTL-trained D+R+D and D+B+D models. The reason for choosing the Calibration method for comparison is that, among all the considered SOTA methods, this particular approach bears the closest resemblance to our proposed LTL learning method in terms of its underlying mechanism. The LTL-trained D+R+D and D+B+D models exhibit minimal accuracy drops of 0.56% and 0.31%, respectively, from the teacher ANN. This suggests the LTL rule is highly effective in producing both rate- and burst-based neural representations. In the same vein of research, the Calibration method [108] is proposed to fine-tune the hidden layer representations in a layer-wise manner, following the principle of rate coding. However, this method neglects the temporal dynamics of spike trains and only conducts training at the spike-train level. Consequently, their model employing D+R+D coding encounters a larger accuracy drop of 1.21%.

To analyze the underlying reason for this performance improvement, we calculate the normalized MSE losses (i.e., Eq. (3.6)) between the convolutional layers of teacher ANN and student SNN, the results are shown in Figure 3.6(a). As expected, the LTL-trained D+R+D model achieves significantly lower MSE losses than those of the Calibration model. This finding suggests that fine-tuning the temporal dynamics of each hidden layer is crucial for achieving the desired neural representations. Fur-

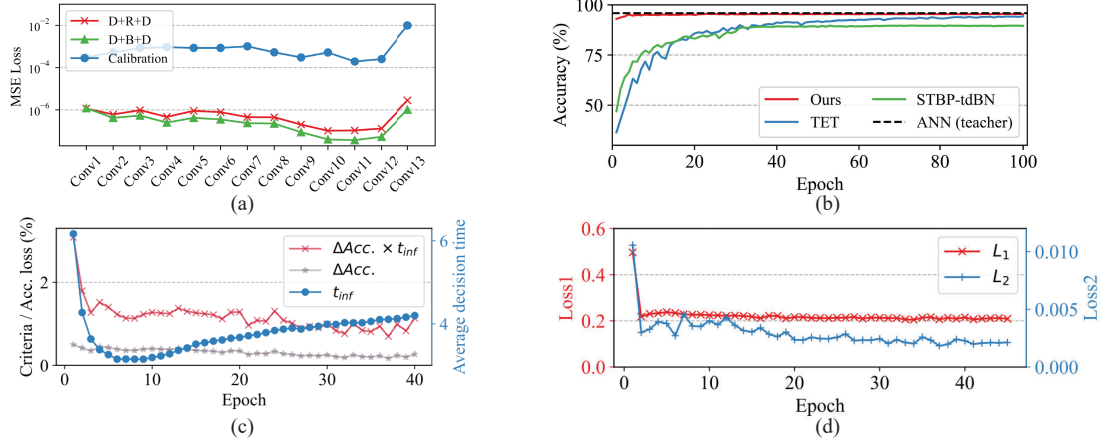


**Figure 3.6: Learning dynamics of the proposed layer-wise learning methods.** (a) Performance validation of our proposed LTL learning method in achieving desired burst coding across hidden layers. We compare our method with the Calibration method, which bears the closest similarity to ours in terms of the underlying mechanism. (b) Learning curves comparison among our layer-wise training method LTL, direct training method STBP-tdBN, and TET, highlighting the faster convergence of our LTL training method. (c) Illustration of learning dynamics during TTFS learning.  $t_{inf}$  refers to the average inference time,  $\Delta Acc.$  is the accuracy drop from baseline ANN to SNN, and  $\Delta Acc. \times t_{inf}$  is the evaluation criterion used in our study to assess the overall model performance. (d) The evolution of the two training losses used during TTFS learning, which govern the learning dynamics observed in (c).

thermore, the LTL-trained D+B+D model outperforms the D+R+D model slightly, indicating burst coding can better approximate the neural representations of teacher ANN than rate coding.

Additionally, by leveraging the knowledge from teacher ANN, the layer-wise LTL rule can achieve faster training convergence than other end-to-end training rules. To illustrate this, we compare the learning curves of the LTL rule with the STBP-tdBN method [31], and the TET rule [32]. These two methods are end-to-end SNN training methods that are directly comparable to our method, while other methods mentioned employ ANN-to-SNN conversion methods to obtain network parameters from a pre-trained ANN. As depicted in Figure 3.6(b), our model reaches the accuracy of 95.19%

rapidly within only 5 training epochs, whereas the STBP-tdBN model converges after 50 epochs and the TET model does not converge even at epoch 100. Beyond the early decision-making capability as discussed earlier, our hybrid neural coding and learning framework also significantly reduces the cost of training and deployment of SNNs.

In general, the quality of the decision improves when a longer inference time is involved in an SNN. To illuminate how the proposed TTFS learning rule effectively reduces the latency without compromising classification accuracy, we plot the learning curves of the TTFS classifier in Figures 3.6(c, d). Both accuracy and inference latency are objectives to be optimized during training. Specifically, we employ  $\Delta\text{Acc.} \times t_{inf}$  as the evaluation criteria. As shown in Figure 3.6(c), the average decision time is sharply advanced at the beginning, accompanied by a degree of accuracy loss due to overfitting. The learning dynamics during this period are mainly governed by the long-term potentiation (LTP) effect of loss function  $\mathcal{L}_1$  as described in Figure 3.6(d). Fortunately, the training process transitions to be dominated by  $\mathcal{L}_2$  after the convergence of  $\mathcal{L}_1$ , which improves the classification accuracy by slightly delaying wrong decisions, allowing more input evidence to accumulate. The learning dynamics at these two stages perfectly align with the two objectives specified in the TTFS learning rule, thereby ensuring an early yet accurate decision is made.

### 3.3.3 Superior Energy Efficiency with Hybrid Coding

In the previous sections, we demonstrated that the proposed hybrid neural coding and learning framework can achieve rapid decision-making. Here, we analyze whether this attribute can further contribute to energy savings. We adhere to a common practice to assess the energy consumption of SNNs, which was initially proposed by IBM TrueNorth neuromorphic chip in their seminal Science paper [157]. It has since become the prevailing standard in the neuromorphic computing community for benchmarking the energy efficiency of different SNN and ANN models [39, 46, 158].

This method focuses on quantifying the energy consumption associated with the basic operations in SNNs and ANNs, i.e., synaptic operations (SynOps). It provides a convenient and reliable way to estimate and compare energy consumption across different models, thus enabling equitable energy performance evaluations. To this end, we calculate the SynOps ratio between equal-structured ANN and SNN by multiply-and-accumulate (MAC) and accumulate (AC) operations, respectively. The total SynOps required in ANN models can be calculated as:

$$\text{SynOps(ANN)} = \sum_l^L f_{in}^l N^l, \quad (3.14)$$

where  $L$  is the total number of layers,  $f_{in}^l$  denotes the fan-in connections to layer  $l$  from the previous layer, and  $N^l$  is the total number of neurons in layer  $l$ . As SNN models have an additional temporal dimension and only operate when spikes occur, their total SynOps calculation is different from their ANN counterparts. Hence, the total SynOps in SNN models are formulated as below:

$$\text{SynOps(SNN)} = \sum_t^T \sum_l^{L-1} \sum_j^{N^l} f_{out,j}^l S_j^l[t], \quad (3.15)$$

where  $T$  is the total inference time,  $f_{out}^l$  represents the fan-out connections from neuron  $j$  in layer  $l$  to the next layer, and  $S_j^l[t]$  denotes the spike of neuron  $j$ .

To facilitate a fair comparison with other SOTA SNN models, we find recent works whose codes are publicly available and reproduce their results on the CIFAR-10 dataset using VGG-16 architecture. Particularly, we report the SynOps ratios at the timestep when a comparable classification accuracy has been achieved. As the results summarised in Table 3.2, our model achieves the lowest SynOps ratio of 0.49, which is at least  $3.90\times$  lower than the other SOTA models. Notably, the total SynOps required by our model is much lower than that of the conversion method QCFS, which achieves a SynOps ratio of 1.91 at  $T = 16$ . The observed discrepancy in energy consumption further validates our initial findings, suggesting that the reduction in inference latency is not primarily due to the compressed time window offered by

**Table 3.2: Comparison of energy efficiency with other SNN implementations when achieving a comparable classification accuracy.**  $\Delta\text{Acc.}$ : the accuracy difference between equal-structured ANN and SNN. **Ratio**:  $\text{SynOps}(\text{SNN})/\text{SynOps}(\text{ANN})$ . **Energy Saving**: Estimated energy saving of SNN against ANN using 32-bit float operations (32-bit integer operations).

| Model                       | SNN Acc. (%) | $\Delta\text{Acc.}$ (%) | Avg. Inference Time | Ratio       | Energy Saving        |
|-----------------------------|--------------|-------------------------|---------------------|-------------|----------------------|
| Calibration [108]           | 95.04        | -0.65                   | 64                  | 2.10        | 2.43 (15.24)         |
| QCFS [28]                   | 93.43        | -0.15                   | 16                  | 1.91        | 2.68 (11.94)         |
| Burst+LIPooling [141]       | 93.71        | -0.40                   | 64                  | 4.81        | 1.06 (6.65)          |
| TET [32]                    | 94.94        | -                       | 6                   | 2.06        | 2.48 (15.53)         |
| <b>Ours (Hybrid Coding)</b> | <b>95.66</b> | <b>-0.17</b>            | <b>4.87</b>         | <b>0.49</b> | <b>10.43 (65.31)</b> |

burst coding. Instead, burst coding allows important information to be transmitted earlier, allowing early and efficient decisions made by TTFS coding. Surprisingly, the SynOps ratio of the TET directly trained model is significantly higher than ours, even at a comparable inference time. We attribute this to the activity regularization loss introduced in TET training. Specifically, this loss term promotes the stable generation of spikes over time. Once removing this loss term, the models tend to spike more frequently during the final few timesteps, which in turn reduces the overall firing rate.

Empirical evidence from a recent study [159] on 45 nm CMOS technology reveals that the 32-bit float MAC and AC operations consume 4.6 pJ and 0.9 pJ energy, respectively. As such, our SNN model can yield an order of magnitude energy saving over its ANN counterpart. The saving can be further boosted to  $65.31\times$  when the cheaper 32-bit integer operations are used (3.1 pJ and 0.1 pJ for corresponding MAC and AC operations).

In practical neuromorphic systems, a substantial portion of energy consumption also comes from memory access and spike-based I/O rather than arithmetic operations alone. This provides an additional perspective for understanding the efficiency characteristics of our model. Our proposed TTFS coding mechanism offers a structural

**Table 3.3: Comparison of sound localization accuracy, MAE, and inference time on the SLoClas dataset.**

| Dataset | Method                      | SNN | Parameters | MAE (°)     | Accuracy (%) | Avg. Inference Time |
|---------|-----------------------------|-----|------------|-------------|--------------|---------------------|
| SLoClas | GCC-PHAT-CNN [153]          | No  | 4.17M      | 4.39        | 86.94        | -                   |
|         | SSLNN [160]*                | No  | 1.60M      | 3.13        | 85.13        | -                   |
|         | SELDnet [161]*              | No  | 1.68M      | 1.78        | 88.24        | -                   |
|         | EINV2 [162]*                | No  | 1.63M      | 0.98        | 94.64        | -                   |
|         | SRP-DNN [163]*              | No  | 1.64M      | 0.96        | 94.12        | -                   |
|         | FN-SSL [164]*               | No  | 1.68M      | 0.63        | 95.40        | -                   |
|         | MTPC-CSNN [44]*             | Yes | 1.61M      | 1.23        | 93.95        | 4                   |
|         | MTPC-CSNN [44]*             | Yes | 1.61M      | 1.02        | 94.72        | 8                   |
|         | MTPC-RSNN [44]*             | Yes | 1.67M      | 1.48        | 94.30        | 51                  |
|         | <b>Ours (Hybrid Coding)</b> | Yes | 1.61M      | <b>0.60</b> | <b>95.61</b> | <b>4.37</b>         |

\* Our reproduced results are based on publicly available source code.

advantage in this context: once a neuron emits its first spike at timestep  $t_f$ , it can terminate subsequent updates and avoid further memory operations for the remaining  $T - t_f$  timesteps. Under this mechanism, the I/O energy cost can be theoretically reduced in proportion to  $t_f/T$ . Therefore, although the present analysis is based on theoretical arithmetic-level energy estimation, the reduction in both computation and I/O activities indicates that the hybrid coding framework has the potential to retain its energy benefits on neuromorphic hardware as well, making it suitable for energy-efficient deployment in event-driven systems.

### 3.3.4 Precise, Rapid, and Robust Sound Localization

In this subsection, we evaluate the proposed hybrid neural coding and learning framework on the SLoClas dataset [153], which is designed for real-world sound localization and classification tasks. Particularly, we comprehensively compare the hybrid coding SNN against other SOTA models, focusing on localization precision, inference latency, energy consumption, and noise robustness.

To provide a comprehensive assessment of the localization precision for our proposed

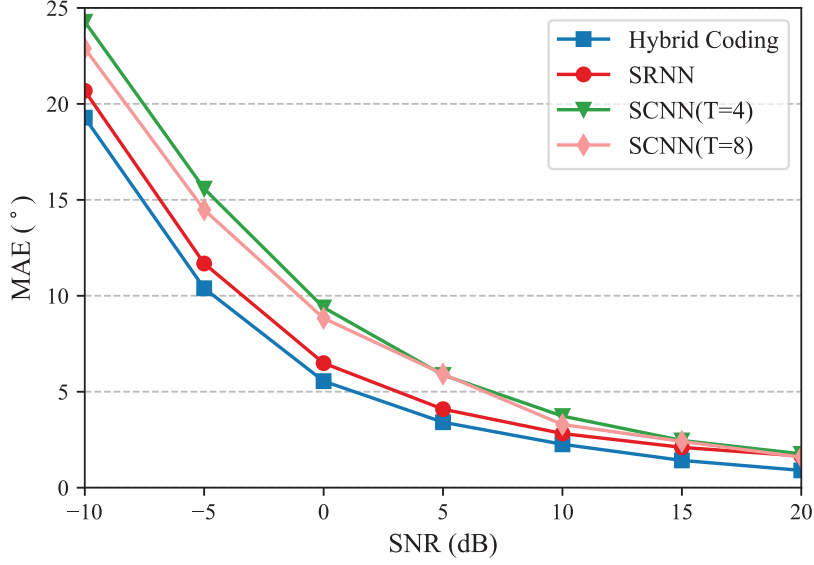


**Figure 3.7: Comparison of localization precision under different SNRs among different SNN models.**

model and other SOTA methods, we report both the mean absolute error (MAE) and the classification accuracy defined as follows [153]:

$$\begin{aligned} \text{MAE } (^{\circ}) &= \frac{1}{N_t} \sum_{i=1}^{N_t} |\hat{\theta}_i - \theta_i|, \\ \text{Acc. } (\%) &= \frac{1}{N_t} \sum_{i=1}^{N_t} (|\hat{\theta}_i - \theta_i| < \eta), \end{aligned} \tag{3.16}$$

where  $\hat{\theta}_i$  and  $\theta_i$  are the estimated and the ground truth azimuth angle of sample  $i$ .  $\eta$  denotes the acceptable absolute error in determining whether a sample is correctly classified.  $N_t$  is the total number of test samples. Given the raw audio is recorded at a resolution of  $5^{\circ}$ , we maintain consistency by setting  $\eta$  to  $2.5^{\circ}$ . Consequently,  $\hat{\theta}_i$  is rounded to the nearest 5-degree increment when calculating classification accuracy.

As presented in Table 3.3, our hybrid coding SNN achieves an MAE of  $0.60^{\circ}$  with an average inference time of only 4.37 timesteps, outperforming all other SOTA SNN models while requiring comparable or shorter inference time. The localization precision is also competitive with other recently introduced ANN models.

**Table 3.4: Comparison of MAE, inference time, and SynOps among different neural coding combinations.**

| Model | MAE (°) | Avg. Inference Time | SynOps (Millions) |
|-------|---------|---------------------|-------------------|
| ANN   | 1.12    | -                   | 5.15              |
| P+R+D | 0.83    | 16                  | 2.06              |
| P+B+D | 0.61    | 16                  | 10.70             |
| P+R+T | 0.82    | 7.62                | 0.94              |
| P+B+T | 0.60    | 4.37                | 2.86              |

Furthermore, the noise robustness of the proposed model is evaluated against other SNN models. The directional background noise fragments provided in the SLoClas dataset are randomly selected and added to the test audio clips at different signal-to-noise ratios (SNRs). As demonstrated in Figure 3.7, our hybrid coding SNN consistently demonstrates higher robustness against background noise, regardless of the levels of noise present.

### 3.3.5 Ablation Study of Hybrid Coding Design

To systematically investigate the impact of various hybrid coding combinations for the sound localization task, we have conducted a series of ablation studies. Specifically, we thoroughly analyze the four most promising combinations: P+R+D, P+B+D, P+R+T, and P+B+T.

#### Localization Precision, Latency, and Energy Consumption

We first evaluate the MAE, inference time, and SynOps of different hybrid coding designs, as demonstrated in Table 3.4. To investigate the impact of the output decoding scheme on network performance, we first conduct a comparative analysis between P+B+T and P+B+D, as well as between P+R+T and P+R+D. Obviously, direct

decoding and TTFS coding yield nearly identical MAEs, whereas TTFS coding only requires about one-third to one-half of the average inference time. These results further support our claim that TTFS coding can lead to reduced inference latency and energy consumption while maintaining high precision. Intriguingly, the ratios of energy efficiency and inference time between the two output coding schemes exhibit consistent values, irrespective of what coding schemes are used in the hidden layers. This observation implies that the reduced inference time plays a pivotal role in total energy saving.

Furthermore, the results also offer valuable insights into the impact of hidden layer coding schemes. Burst coding outperforms rate coding by reducing 0.22° MAE while accelerating decision-making by 3.25 timesteps. However, burst coding may result in increased spike firing rates, in turn leading to higher energy consumption. To optimize the network performance, the choice between burst coding and rate coding is contingent upon the specific requirements of the task at hand. For instance, if the priority is given to energy efficiency, the P+R+T design could be the preferred choice. Alternatively, if minimizing MAE and latency are the primary concerns, the P+B+T design becomes an unquestionably superior choice. Notably, even though it demands more energy than the P+R+T design, it still only requires half the energy consumption of a competitive ANN model, striking an optimal balance between precision, latency, and energy consumption.

## Environmental Noise

In order to investigate the impact of hybrid coding designs on noise robustness, we generate noise-corrupted test samples using two distinct forms of environmental noise, namely factory noise and babble noise. All models are trained on clean train samples and subsequently evaluated on noise-corrupted test samples with varying SNRs. The results of different hybrid coding designs are presented in Figure 3.8. In general, the

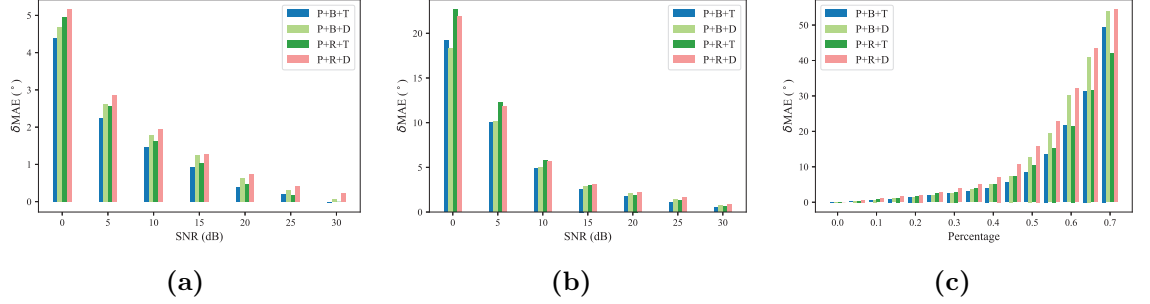


**Figure 3.8: The sound localization precision comparison under different noise conditions:** (a) babble noise, (b) factory noise, (c) neuronal noise.  $\delta\text{MAE}$  is the MAE discrepancy between noisy and clean conditions.

P+B+T design demonstrates the highest noise robustness under most conditions. This notable advancement can be attributed to the fact that TTFS coding is less susceptible to cumulative noise, as it only requires about one-third of the time for decision-making. However, TTFS coding may also yield adverse effects when environmental noise significantly impairs information integrity, as it can only leverage limited evidence due to the reduction of inference time. Furthermore, our results suggest that hybrid coding SNNs consistently perform better under babble noise than under factory noise, owing to the fact that the latter has larger frequency variations.

### Neuronal Noise

Furthermore, we analyze the robustness of hybrid coding designs against neuronal noise, which is known to be ubiquitous in human neural systems. Investigating the network performance under neuronal noise provides an opportunity to identify the most promising design that closely mimics human auditory systems. The neuronal noise is simulated by introducing spike deletion operations, which means in each timestep, a certain fraction of hidden neurons fail to generate output spikes at a given probability. The results are visualized as a function of deletion rate in Figure 3.8(c). It indicates that TTFS coding is more reliable than direct decoding under

neuronal noise conditions, which is consistent with our observations on environmental noise. Interestingly, when a large percentage of information fails to transfer ( $>60\%$ ), the P+R+T design surpasses the P+B+T design in performance. This observation highlights the inherent advantage of rate coding, whose spikes are more informative than burst coding. Therefore, it can preserve more information with fewer spike counts.

### 3.4 Discussion and Conclusion

This chapter enriches the information representation strategies of spiking neuron models by introducing the hybrid neural coding and learning framework, which integrates diverse biological coding schemes to enhance the fidelity, efficiency, robustness, and adaptability of neural representations across various pattern recognition tasks. In particular, we demonstrate that by holistically designing SNNs to encompass hybrid neural coding schemes, the benefits of different coding strategies can be leveraged to achieve optimal performance under diverse task requirements and environmental conditions. To overcome the challenges associated with training hybrid coding SNNs, we propose a layer-wise learning method that is both highly effective and efficient in producing the desired neural representations compared to existing SNN learning methods. Extensive experiments show that SOTA image classification accuracies and sound localization precision can be achieved with significantly improved inference speed, energy efficiency, and noise robustness compared to conventional SNNs.

While our study primarily examines image classification and sound localization tasks, the design principles and training methods proposed in this chapter can be extended to other pattern recognition tasks and diverse environmental conditions. Notably, the successful implementation of our hybrid coding approach has facilitated a deeper understanding of the strengths and limitations of various neural coding schemes as well as hybrid spiking neuron models. In future research, we aim to develop a neu-

ral coding search method that can automatically identify the optimal hybrid neural coding design for each spiking neuron according to task demands and environmental conditions, thereby reducing the required level of expertise.

Remarkably, the proposed hybrid coding strategy is highly compatible with contemporary spiking transformer architectures. By leveraging burst-based computation within attention and feedforward blocks, and employing early decision-making through TTFS coding in the output layer, our model holds strong potential for reducing inference latency and energy cost in such architectures. In addition, the hybrid coding framework can be efficiently deployed on existing neuromorphic chips. Specifically, the phase, burst, and TTFS neurons used in the framework can be mapped to neuromorphic primitives through simple yet effective hardware mechanisms. These include the use of AND logic gates with binary inputs and outputs to implement coincidence-based phase detection, programmable packet payloads to enable burst transmission, and event-driven routing schemes to support TTFS-based early exiting. In future work, we plan to further investigate the scalability and hardware feasibility of the proposed hybrid coding framework.

## Chapter 4

# A Parallel Multi-compartment Spiking Neuron for Multi-scale Temporal Processing

This chapter advances spiking neuron models with enhanced structural complexity by introducing the PMSN, which improves expressiveness while supporting efficient parallel computation of spiking neurons. In contrast to conventional spiking neuron models that abstract neurons as indivisible point units with homogeneous dynamics, PMSN incorporates multiple interacting substructures to enable effective information integration across diverse timescales. Additionally, to address the computational cost introduced by the increased complexity of neuronal dynamics, we further introduce two parallelization techniques that allow efficient computation across time. Extensive experiments across various pattern recognition benchmarks demonstrate the superiority of PMSN, which consistently outperforms other SOTA spiking neuron models by providing significant training acceleration while striking a favorable trade-off between accuracy and computational cost. These results present PMSN as a promising solution for harnessing the computational advantages of detailed spiking neuron models

in neuromorphic pattern recognition.

## 4.1 Introduction

SNNs have been proposed as one of the most representative brain-inspired computational models, aiming to mimic the efficient spatiotemporal information processing in the brain [17]. While SNNs have gained increasing attention in recent years, their main applications have been primarily confined to computer vision tasks that involve limited temporal dynamics, such as classifying static images or data collected from DVS data with artificially induced saccadic motion [121, 122]. However, real-world pattern recognition scenarios are often more challenging as they involve sensory signals with information that varies across different timescales. For instance, speech recognition tasks often necessitate models to establish dependencies across various timescales, encompassing phonemes, words, and sentences. While several algorithms have been developed to enhance the temporal processing capacity of SNNs [35, 46, 158, 165], most of them still struggle to establish diverse scales of temporal representations, resulting in poor performance in complex temporal processing tasks.

Considering the remarkable temporal processing capabilities observed in biological neurons, it is crucial to carefully examine their computational mechanisms. In the context of large-scale SNNs, the majority of existing spiking neurons are modeled as single-compartment systems, such as the LIF model [51]. In these single-compartment models, the neuronal dynamics are simplified to first-order dynamics of a single state variable, the membrane potential, disregarding any interactions among substructures within a single neuron. This simplification reduces computational complexity but limits their ability to generate complex neuronal dynamics.

In contrast, real biological neurons can be more accurately modeled by the multi-compartment neuron models [53, 166, 167, 168], which divide a single neuron into sev-

eral interconnected subunits with interactions among them. Extensive neuroscience experiments have evidenced the important roles of these inter-compartment dynamics in temporal processing. For example, the nonlinear interactions among ion channels endow neurons with significant computational power for processing temporal signals [4, 169]. Additionally, the interaction among coupled dendritic components acts as temporal filters for signals traveling from dendrites to the soma, thereby facilitating the temporal sequence detection [53, 64]. While detailed multi-compartment models show great potential in temporal processing, they are computationally expensive due to the intricate anatomical structures of dendritic trees and high-dimensional ionic properties. Consequently, they are not well-suited for constructing large-scale SNNs to tackle real-world pattern recognition tasks. Therefore, there is a pressing need to develop an efficient spiking neuron model that strikes a balance between computational complexity and the ability to capture the valuable interaction among different substructures of a biological neuron.

Recently, several simplified compartmental spiking neuron models have been specifically designed for deep SNNs. For instance, a two-compartment model was introduced to simulate double-exponential threshold decay [71], and a dendritic neuron model was proposed to endow multiple dendritic compartments with heterogeneous decay time constants [47]. Furthermore, inspired by the well-known Pinsky-Rinzel (P-R) pyramidal neuron located in the CA3 region of the hippocampus, a two-compartment LIF model was proposed, which divides a single neuron into dendritic and soma compartments [21].

Nevertheless, there are three significant challenges that remain to be tackled. Firstly, these models overlook the valuable recurrent interactions among neuronal compartments, which play a crucial role in integrating temporal information across multiple timescales in biological neurons [5, 170]. As a result, the computational expressiveness of these models is limited, constraining the complexity of temporal dynamics they can effectively capture. Secondly, these meticulously hand-crafted models are inherently

constrained by a predetermined number of compartments. While increasing this number allows for richer structural complexity and a broader range of temporal dynamics, reducing it yields a more compact and computationally efficient representation. However, the inability to adjust this abstraction level limits these models' adaptability to tasks with varying temporal complexity and computational demands. Thirdly, compared to single-compartment models, the increased temporal dynamic complexity in these models results in significantly slower training processes when using the BPTT algorithm. Consequently, these models demonstrate limited scalability to larger networks for tackling challenging real-world temporal processing tasks, particularly those involving long sequences.

To tackle these challenges, we propose a generalized multi-compartment spiking neuron model for SNNs that integrates essential recurrent interactions among interconnected compartments. Moreover, the number of compartments can be flexibly adjusted in the proposed model, allowing for adaptation to the different temporal complexities required in real-world tasks. Furthermore, considering the significant constraint on training speed caused by richer neuronal dynamics in the proposed model, we introduce two temporal parallelization techniques to accelerate the training process. First, for the linear recurrence associated with the neuron charging process, we draw inspiration from the recent studies in deep learning showing that recurrent models with linear recurrence can be effectively parallelized in the temporal domain [171, 172, 173]. Specifically, we design the neuron charging process to function as a linear time-invariant (LTI) system. Second, for the nonlinear recurrence associated with the neuron firing and reset process, we introduce a novel reset strategy that decouples the temporal dependencies. Altogether, these strategies enable efficient parallel training of our model over time.

The proposed design methodologies have led to the development of a novel PMSN model as illustrated in Figure 4.1. The PMSN model effectively captures multi-scale temporal information through the interaction among neuronal compartments.

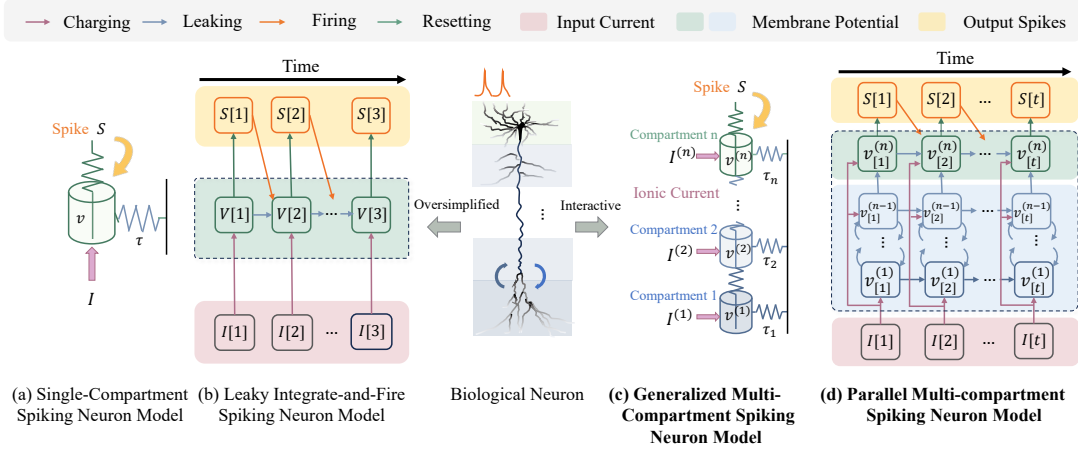


**Figure 4.1: Comparison of neuronal structure and dynamics between the single-compartment model, biological neurons, and the proposed multi-compartment spiking neuron model.** (a, b) The widely used Leaky Integrate-and-Fire model simplifies the biological neurons into a single unit and ignores the interaction among neuronal substructures, resulting in deficiencies in multi-scale temporal processing and slow training speed. (c) In contrast, the detailed morphologies and electrophysical properties of biological neurons improve their computational power in interactive ways, which is crucial for temporal information processing. Drawing inspiration from these, we propose a generalized multi-compartment spiking neuron model that divides a single neuron into a variable number of interconnected subunits with heterogeneous temporal properties and interactive dynamics. (d) Our proposed Parallel Multi-compartment Spiking Neuron model further extends the compartmental structure in (c). It not only supports rich neuronal dynamics essential for temporal processing, but also facilitates efficient parallel training across time.

Both theoretical analysis and dynamics visualizations are provided to substantiate the efficacy of the PMSN model in establishing temporal dependencies across various timescales. Moreover, the proposed PMSN model offers a significant improvement in training speed, particularly when deployed on GPU-accelerated ML frameworks. The main contributions of this chapter are summarized as follows:

- To enhance the structural complexity of spiking neuron models, this chapter proposes a generalized multi-compartment spiking neuron model that incorporates valuable interactions among different neuronal compartments. The number of compartments in this model can be flexibly adjusted to represent temporal information

across diverse timescales.

- This chapter develops two temporal parallelization techniques for the proposed model, resulting in a remarkable training acceleration of over  $10\times$  times on GPU-accelerated ML frameworks.
- The comprehensive experimental studies on numerous benchmarks demonstrate that the proposed model not only showcases superior temporal processing capacity but also offers a favorable computational cost when compared to single-compartment models.

The remainder of this chapter is organized as follows: Section 4.2 introduces the PMSN model, outlining its generalized multi-compartment neuronal dynamics, temporal parallel computation strategies, and mechanisms for effective gradient propagation. Section 4.3 demonstrates the performance of PMSN on multiscale spatiotemporal processing and static image classification tasks, highlighting its superior accuracy, accelerated training speed, and comparable computational cost. Finally, Section 4.4 concludes the chapter.

## 4.2 PMSN: A Parallel Multi-compartment Spiking Neuron

In this section, we present the proposed PMSN model, which integrates recurrent interactions among interconnected compartments to facilitate effective temporal integration while enabling efficient parallel computation across time. We begin by revisiting the inherent limitations of single-compartment spiking neuron models in processing temporal signals, which motivate the development of our designs. We then introduce a generalized multi-compartment neuron that captures the structural complexities of biological neurons, forming the foundation of PMSN’s neuronal dynamics.

Building on this, we further introduce two parallel computation strategies that ensure scalable and efficient training of PMSN. Finally, we demonstrate how PMSN achieves effective temporal gradient propagation, which is essential for establishing multi-scale temporal dependencies.

### 4.2.1 Revisit: Single-compartment Spiking Neuron Model Struggles to Process Temporal Signals

In this subsection, we revisit the LIF model as a representative of single-compartment spiking neuron models. We first introduce basic concepts and then discuss the major challenges associated with processing temporal signals using this model. As illustrated in Figure 4.1, the temporal dynamics of the LIF model can be formulated as:

$$\begin{aligned} \text{Leak \& Charge: } \quad & \frac{dV(t)}{dt} = -\frac{1}{\tau_m}(V(t) - V_{rest}) + I(t), \\ \text{Fire \& Reset: } \quad & \text{if } V(t) \geq \theta, S(t) = 1, V(t) \rightarrow V(t) - \theta. \end{aligned} \tag{4.1}$$

During the membrane potential leaky and charging phase, the information contained in the input current  $I(t)$  is integrated into the membrane potential  $V$  and further undergoes decay at a rate governed by  $\tau_m$ .  $V_{rest}$  is the resting potential. Once  $V(t)$  exceeds the firing threshold  $\theta$ , an output spike  $S(t)$  will be generated and transmitted to subsequent neurons. Following the spike generation, the membrane potential will be reset. In practice, the above continuous-time formulation is typically discretized using the Euler method as

$$\begin{cases} V[t] = \alpha V[t-1] + I[t] - \theta S[t-1], \\ S[t] = H(V[t] - \theta), \end{cases} \tag{4.2}$$

where  $H(\cdot)$  is the Heaviside function and  $\alpha = \exp(-\frac{dt}{\tau_m})$  is the discrete-time decay factor.

Despite its promising results in tasks involving limited temporal context, LIF neurons encounter the following two challenges when dealing with long sequences. Firstly, this

model faces challenges in retaining information over an extended time period, primarily due to its inadequate representation of neuronal dynamics. Specifically, the LIF model overlooks the computational role played by interaction among different neuronal substructures. As a result, the membrane potential  $V$  is the only state variable that can integrate and store temporal information. Unfortunately, this state variable is subject to exponential decay and reset, which hinders the establishment of long-term temporal dependencies. Secondly, its training time increases proportionally with the sequence length, making it inefficient for tasks that involve long sequences. This slow training speed arises from the nonlinear neuronal updates in the LIF model, which must be computed step-by-step in the temporal dimension and leads to underutilization of the full potential of GPU acceleration. Specifically, the membrane potential update of  $V[t]$  depends on the output spike  $S[t - 1]$  from the preceding timestep, which is generated by applying a nonlinear Heaviside activation to  $V[t - 1]$ . As a result,  $V[t]$  is not available until  $V[t - 1]$  has been fully processed. This time-coupled relationship prevents the membrane potential dynamics from being unfolded across time, leading to difficulties in temporal parallelization.

### 4.2.2 A Generalized Multi-compartment Spiking Neuron with Interactive Dynamics

In this subsection, we first briefly introduce the multi-compartment models that have been widely used in neuroscience studies. Drawing inspiration from them, we propose a generalized multi-compartment spiking neuron model that can strike a favorable balance between the richness of neuronal dynamics and computational cost. Notably, in contrast to commonly used multi-compartment neuron models in deep SNNs, our model retains the crucial recurrent interactions among interconnected compartments. It thereby allows more effective representation and processing of temporal signals across different timescales.

Compartmental models divide a single neuron into interconnected subunits or “compartments”, with specific spatial structures. The interactions among these interconnected compartments lead to the rich neuronal dynamics observed in biological neurons that play a crucial role in sensory processing [174]. For instance, Rall’s cable theory [64] suggests each dendrite can be mathematically modeled as a series of interconnected compartments resembling a cable structure. However, given the complex anatomical morphology of dendrite trees, these models pose significant computational challenges and are impractical for large-scale simulations. To reduce the modeling difficulty, quantitative models with reduced compartment numbers and structure complexity have been explored. For instance, the neuronal activities of CA3 pyramidal neurons can be modeled by a 19-compartment cable model, wherein dendritic branching is omitted [175]. This model has been further abstracted to a more computationally efficient two-compartment model [176]. In these quantitative models, compartments are restricted to interacting with their adjacent compartments solely, and the dynamics of each compartment can be described as a differential equation of a resistor-capacitance circuit as

$$C_m \frac{dv^{(i)}}{dt} = -I_{Leak}(v^{(i)}) + g_{i-1,i}(v^{(i-1)} - v^{(i)}) + g_{i,i+1}(v^{(i+1)} - v^{(i)}) + I. \quad (4.3)$$

where  $C_m$  is the capacitance of membrane potential,  $I_{Leak}(\cdot)$  represents the leakage current of membrane potential  $v^{(i)}$ ,  $g_{i,j}(\cdot)$  is the coupling conductance between compartments  $i$  and  $j$ , and  $I$  denotes the summation of other voltage-gated ionic currents.

In order to simplify the ionic compartmental dynamics described in Eq. (4.3) and ensure compatibility with existing GPU-accelerated ML frameworks, we next introduce a generalized multi-compartment spiking neuron model with a simplified neuronal hyperparameter setup. This model retains the essential interactions among different substructures of a neuron, which are theoretically evidenced in Section 4.2.4 as playing a crucial role in capturing and integrating temporal features across multiple timescales. Another notable feature of our model is its flexibility, allowing for a vary-

ing number of neuronal compartments  $n$  to suit the complexity of the task at hand. The detailed formulation of our proposed model is given as

$$\begin{cases} \frac{dv^{(1)}(t)}{dt} = -\frac{1}{\tau_1}v^{(1)}(t) + \beta_{2,1}v^{(2)}(t) + \gamma_1 I(t), \\ \frac{dv^{(2)}(t)}{dt} = -\frac{1}{\tau_2}v^{(2)}(t) + \beta_{3,2}v^{(3)}(t) + \beta_{1,2}v^{(1)}(t) + \gamma_2 I(t), \\ \dots \\ \frac{dv^{(n)}(t)}{dt} = -\frac{1}{\tau_n}v^{(n)}(t) + \beta_{n-1,n}v^{(n-1)}(t) + \gamma_n I(t), \end{cases} \quad (4.4)$$

if  $v^{(n)}(t) \geq \theta, \quad S(t) = 1, \quad v^{(n)}(t) \rightarrow v^{(n)}(t) - \theta,$

where  $v^{(i)}$  represents the membrane potential of the compartment  $i$ ,  $\theta$  is the firing threshold, and  $I(t)$  denotes the synaptic current transduced from the output spikes from the preceding layer. Once the membrane potential of the final compartment  $v^{(n)}$  exceeds the threshold  $\theta$ , it triggers an output spike and simultaneously resets  $v^{(n)}$ . The compartmental parameters, including  $\tau_i$ ,  $\gamma_i$ , and  $\beta_{i,j}$ , represent the membrane time constant, input attenuation of the compartment  $i$ , and coupling strength between interconnected compartments  $i$  and  $j$ , respectively.

For the sake of simplicity in representation, we can also formulate these first-order differential equations as an  $n$ -order state space function:

$$\dot{\mathcal{V}}(t) = \begin{bmatrix} -\frac{1}{\tau_1} & \beta_{2,1} & 0 & \dots & 0 \\ \beta_{1,2} & -\frac{1}{\tau_2} & \beta_{3,2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & -\frac{1}{\tau_{n-1}} & \beta_{n,n-1} \\ 0 & 0 & \dots & \beta_{n-1,n} & -\frac{1}{\tau_n} \end{bmatrix} \mathcal{V}(t) + \gamma_n I(t), \quad (4.5)$$

$$S(t) = H(v^{(n)}(t) - \theta), \quad v^{(n)}(t) = v^{(n)}(t) - \theta S(t),$$

where  $\gamma_n = [\gamma_1, \dots, \gamma_n]^T$ ,  $\mathcal{V} = [v^{(1)}, \dots, v^{(n)}]^T$ .

It is important to note that many widely used spiking neuron models can be seen as special cases of our proposed model by adjusting its hyperparameters. For instance, when the number of compartments  $n$  is set to 1 and  $\beta = 0$ , our model is simplified to a standard LIF model. Similarly, when  $n = 2$  and  $\gamma_2 = 0$ , our model degrades into

a TC-LIF model [21]. These examples highlight the generalizability of our model. Moreover, by providing flexibility in increasing the number of compartments, our model surpasses existing spiking neuron models and offers richer neuronal dynamics that are essential for complex temporal processing tasks.

### 4.2.3 Efficient Parallel Computation of Temporal Dynamics

The generalized multi-compartment spiking neuron model introduced earlier requires significantly more training time compared to existing spiking neuron models used in deep SNNs, primarily due to its higher computational complexity. This limitation restricts its practical deployment in long sequence temporal processing scenarios. Thus, it becomes crucial to develop strategies to enable parallel training of this model in time. However, it is not straightforward to employ existing parallel computation techniques [171], due to the involvement of the nonlinear function  $H(\cdot)$  for spike generation and reset processes.

In this subsection, we further introduce two parallel strategies for the hidden and output compartments, yielding the PMSN formulations. To better explain our idea, we divide the total  $n$  compartments into  $n - 1$  hidden compartments  $\mathcal{V}_h$  with linear recurrence and one output compartment  $v^{(n)} = v_s$  with nonlinear firing and reset dynamics. Meanwhile, the feedback from the output to the last hidden compartment  $\beta_{n,n-1}$  is set to 0. The resulting neuronal function is represented as

$$\dot{\mathcal{V}}_h(t) = \begin{bmatrix} -\frac{1}{\tau_1} & \beta_{2,1} & 0 & \cdots & 0 \\ \beta_{1,2} & -\frac{1}{\tau_2} & \beta_{3,2} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{\tau_{n-1}} \end{bmatrix} \mathcal{V}_h(t) + \gamma_{n-1}I(t), \quad (4.6)$$

$$\dot{v}_s(t) = \beta_{n-1,n}v^{(n-1)}(t) - \frac{1}{\tau_n}v_s(t) + \gamma_n I(t) - \theta S(t), \quad (4.7)$$

$$S(t) = H(v_s(t) - \theta). \quad (4.8)$$

Below, we will introduce how these two parallelization techniques unfold the linear recurrent states  $\mathcal{V}_h$ , and decouple the nonlinear temporal dependency issue associated with  $v_s$  to enable parallel computation.

### Parallel Strategy for Hidden Compartments

We first apply the zero-order hold (ZOH) method [177] to discretize the nonlinear continuous system detailed in Eqs. (4.6) and (4.7). Specifically, we utilize a full-rank state transition matrix  $\mathcal{T} \in \mathbb{R}^{(n-1) \times (n-1)}$  to represent the first matrix in Eq. (4.6), and diagonalize this matrix via eigenvalue decomposition  $\mathcal{T} = P\Lambda P^{-1}$ , where  $\Lambda, P \in \mathbb{C}^{(n-1) \times (n-1)}$  denote the diagonal eigenvalue matrix and eigenvector matrix, respectively. We then obtain the following discrete-time formulation:

$$V_h[t] = \bar{\mathcal{T}}V_h[t-1] + \Phi_c I[t], \quad (4.9)$$

$$I_h[t] = \Phi_s V_h[t] + \gamma_n I[t], \quad (4.10)$$

$$v_s[t] = \alpha v_s[t-1] + I_h[t] - \theta S[t-1], \quad (4.11)$$

$$S[t] = H(v_s[t] - \theta),$$

where  $V_h = P^{-1}\mathcal{V}_h$ ,  $\bar{\mathcal{T}} = \exp(\Lambda dt)$ ,  $\Phi_c = \Lambda^{-1}(\exp(\Lambda dt) - I)\phi_c$ , and  $\phi_c = P^{-1}\gamma_{n-1}$ . The term  $I_h[t]$  signifies the total input to the output compartment,  $\Phi_s = [0, \dots, \beta_{n-1,n}]P$ ,  $\alpha = \exp(-\frac{dt}{\tau_n})$ . The  $\Lambda dt$ ,  $\phi_c$ ,  $\gamma_n$ , and  $\Phi_s$  are learnable parameters that govern the neuronal dynamics. It should be noted that the required complex number operations of PMSN are well-supported by existing neuromorphic chips, such as Intel Loihi [178].

The model described in Eq. (4.9) can be seen as an LTI system. It exhibits a linear recurrence that can be unfolded over time like

$$V_h[t] = \sum_{i=0}^t \bar{\mathcal{T}}^{t-i} \Phi_c I[i]. \quad (4.12)$$

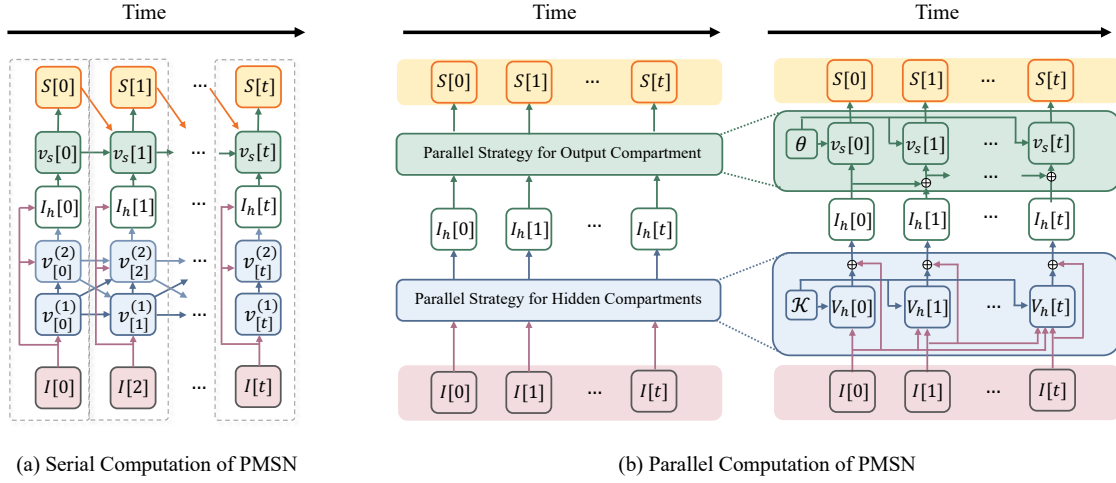


**Figure 4.2: Illustration of the proposed PMSN model and its parallel implementation.**

(a) The PMSN model ( $n = 3$  compartments for demonstration) can be divided into two parts: two hidden compartments with membrane potential  $\mathcal{V}_h = [v^{(1)}, v^{(2)}]$ , and one output compartment with membrane potential  $v_s$ . Due to the intertwined temporal dependencies among these neuron states, calculating their temporal dynamics typically involves serial computation over time, resulting in significant training costs. (b) The proposed parallel solution. **Left:** To accelerate the training speed, two temporal parallel strategies are introduced for the hidden (blue box) and output compartments (green box), respectively. **Right:** For the hidden compartments, the proposed parallel strategy decouples the linear recurrence between  $v^{(1)}$ ,  $v^{(2)}$  into a temporal convolution operation between the input current  $\mathbf{I}_t$  and a kernel  $\mathcal{K}$ , enabling parallel computation of hidden membrane potentials across time. For the output compartment, the parallel strategy employs a novel reset mechanism that leverages the firing threshold  $\theta$  and the accumulated inputs  $I_h$  to compute output spikes at all timesteps simultaneously.

By substituting Eq. (4.12) into Eq. (4.10), we can obtain

$$I_h[t] = \sum_{i=0}^t \Phi_s \bar{\mathcal{T}}^{t-i} \Phi_c I[i] + \gamma_n I[t]. \quad (4.13)$$

Notably, the first term in Eq. (4.13) can be simplified to a convolution form as

$$I_h[t] = \sum_{i=0}^t \mathbf{I}_t[i] \mathcal{K}[t-i] + \gamma_n I[t] = (\mathbf{I}_t * \mathcal{K})[t] + \gamma_n I[t], \quad (4.14)$$

where  $\mathbf{I}_t = \{I[0], \dots, I[t]\}$  denotes the input current set,  $\mathcal{K} = [\Phi_s \bar{\mathcal{T}}^0 \Phi_c, \dots, \Phi_s \bar{\mathcal{T}}^t \Phi_c]$  is the convolution kernel. Consequently, the computation of set  $\mathbf{I}_h = \{I_h[0], \dots, I_h[t]\}$

can be parallelized over time by applying convolution operation on  $\mathbf{I}_t$  and  $\mathcal{K}$ , resulting in

$$\mathbf{I}_h = \mathbf{I}_t * \mathcal{K} + \gamma_n \mathbf{I}_t = \mathcal{F}^{-1}(\mathcal{F}(\mathbf{I}_t) \cdot \mathcal{F}(\mathcal{K})) + \gamma_n \mathbf{I}_t, \quad (4.15)$$

where  $\mathcal{F}$ ,  $\mathcal{F}^{-1}$  represents forward and inverse Fourier Transform, respectively. In this way, we could efficiently compute the membrane potential for the first  $n - 1$  hidden compartments  $V_h$ , and the input for the output compartment  $I_h$  across all timesteps in parallel.

### Parallel Strategy for Output Compartment with Reset

While the dynamics of PMSN within the hidden compartments can be efficiently computed in parallel, the remaining challenge lies in how to achieve parallel computation in the output compartment. However, the output compartment involves a unique spike generation and reset mechanism that introduces a nonlinear dependency across time, posing a major obstacle to parallel computation. (4.11). Specifically, the membrane potential  $v_s[t]$  is updated by subtracting  $\theta S[t - 1]$  at each timestep, where the spike  $S[t - 1]$  is determined by applying a nonlinear Heaviside function to  $v_s[t - 1]$ . This nonlinear dependency between consecutive timesteps prevents  $v_s$  from being unfolded into a closed-form expression and inherently limits the parallel computation of the output compartment across time.

To address this issue, we propose a novel reset mechanism that enables efficient parallel computation during training, while preserving equivalent serial inference. Specifically, we introduce a reset term  $v_r[t]$ , which represents the discharged voltage to be subtracted from the membrane potential  $v_s[t]$  after each spiking. In serial computation,  $v_r$  can be computed sequentially based on the preceding spike as conventional reset mechanisms; In parallel computation, we bypass the nonlinear temporal dependency by directly deriving  $v_r$  across time from the accumulated inputs  $I_h$  in parallel.

The serial computation of the proposed reset mechanism can be formulated as

$$\begin{aligned} v_s[t] &= v_s[t-1] + I_h[t] - v_r[t-1], \\ S[t] &= H(v_s[t] - \theta), \\ v_r[t] &= \theta S[t] \cdot \lfloor v_s[t] \rfloor_\theta, \end{aligned} \tag{4.16}$$

where  $\lfloor x \rfloor_\theta := \theta \lfloor x/\theta \rfloor$  signifies the floor division by  $\theta$ . Compared to the original reset-by-subtraction mechanism in Eq. (4.11), the proposed formulation introduces two key modifications. First, after each spike, the potential  $v_s[t]$  is reset by subtracting the reset term  $v_r[t]$ , which is defined as an integer multiple of the threshold  $\theta$  determined by  $\lfloor v_s[t] \rfloor_\theta$ . This ensures the membrane potential is reset to a subthreshold level after each spiking, more closely mimicking the repolarization process observed in biological neurons. Second, to ensure equivalence between the serial and parallel formulations, the input current  $I_h$  is passed through a ReLU function to ensure non-negativity before being injected into the output compartment. Consequently, the iterative dynamics in Eq. (4.16) can be unfolded across time into a temporally parallel form as

$$\begin{aligned} \sum_{i=0}^{t-1} v_r[i] &= \theta \lfloor \sum_{i=0}^{t-1} I_h[i] \rfloor_\theta, \\ v_s[t] &= \sum_{i=0}^t I_h[i] - \theta \lfloor \sum_{i=0}^{t-1} I_h[i] \rfloor_\theta, \end{aligned} \tag{4.17}$$

where the total reset voltage  $\sum_{i=0}^{t-1} v_r[i]$  can be directly computed by flooring the accumulated input current  $I_h$  across time. The set  $\mathbf{I}_{h,t} = \{I_h[0], \dots, \sum_{i=0}^t I_h[i]\}$ , representing the cumulative input current to the output compartment over time, can be efficiently computed using the parallel prefix sum (scan) algorithm [179]. Therefore, we could effectively derive the output spike train set  $\mathbf{S}_t = \{S[0], \dots, S[t]\}$  based on the obtained membrane potential set  $\mathbf{v}_{s,t} = \{v_s[0], \dots, v_s[t]\}$  as

$$\mathbf{S}_t = H(\mathbf{v}_{s,t}) = H(\mathbf{I}_{h,t} - \theta \lfloor \mathbf{I}_{h,t-1} \rfloor_\theta). \tag{4.18}$$

Notably, our proposed reset mechanism for the output compartment can be easily generalized to other non-leaky single-compartment models, thereby enabling parallel training for a broader spectrum of spiking neuron models.

#### 4.2.4 Effective Temporal Gradient Propagation

Here, we provide a theoretical analysis to explain how the proposed PMSN model enables effective temporal dependency establishment, thereby facilitating multi-scale temporal processing.

To overcome the gradient discontinuity introduced by the Heaviside and floor functions in the proposed reset operation, we employ surrogate gradient methods [32, 149] to facilitate gradient backpropagation in Eq. (4.18), yielding a well-defined gradient flow for the PMSN model:

$$\begin{aligned}
 \Delta \mathcal{W}^l &\propto \frac{\partial \mathcal{L}}{\partial \mathcal{W}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{I}^l[t]} \mathbf{S}^{l-1}[t], \quad \Delta \mathbf{b}^l \propto \frac{\partial \mathcal{L}}{\partial \mathbf{b}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{I}^l[t]}, \\
 \frac{\partial \mathcal{L}}{\partial \mathbf{I}^l[t]} &= \sum_{i=t}^T \frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[i]} \frac{\partial \mathbf{S}^l[i]}{\partial \mathbf{v}_s^l[i]} \frac{\partial \mathbf{v}_s^l[i]}{\partial \mathbf{I}^l[t]} + \sum_{i=t}^{T-1} \frac{\partial \mathcal{L}}{\partial \mathbf{v}_s^l[i+1]} \frac{\partial \mathbf{v}_s^l[i+1]}{\partial \mathbf{v}_s^l[i]} \frac{\partial \mathbf{v}_s^l[i]}{\partial \mathbf{I}^l[t]} \\
 &= \underbrace{\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} g'[t] \gamma_n}_{\text{Spatial}} + \underbrace{\sum_{i=t}^T \frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[i]} g'[i] \Phi_s \bar{\mathcal{T}}^{i-t} \Phi_c}_{\text{Temporal}},
 \end{aligned} \tag{4.19}$$

where  $g'[t] \approx \frac{\partial \mathbf{S}^l[t]}{\partial \mathbf{v}_s^l[t]}$  is the surrogate gradient function,  $\mathcal{L}$  is the loss function, and  $\mathcal{W}^l$ ,  $\mathbf{b}^l$  refer to synaptic weight and bias between layer  $l-1$  and  $l$ , respectively. The first term represents the spatial credit assignment between consecutive network layers, where the gradient propagates from shallow to deeper layers to facilitate hierarchical learning. The second term corresponds to the temporal credit assignment, where gradients flow backward from later to earlier timesteps, facilitating the learning of temporal dependencies across time.

Note that the temporal credit assignment term is governed by the diagonal matrix  $\bar{\mathcal{T}} = \text{diag}(\lambda_1, \dots, \lambda_{n-1})$ , where each complex-valued  $\lambda_i$  encodes both the decaying rate and interaction strength of temporal information. This offers two key advantages of the proposed PMSN model: Firstly, PMSN can capture temporal information across a wide range of timescales. Specifically, the real part of each  $\lambda_i$  determines the membrane potential decaying rate within each compartment, allowing different com-

partments to retain information over varying time scales. Meanwhile, the imaginary part of  $\lambda_i$  modulates interaction strength between compartments, enabling flexible integration of multi-scale temporal information. Secondly, PMSN allows for stable and efficient gradient propagation to earlier timesteps, highlighting its superiority in capturing long-range dependencies. In contrast to conventional SNNs, where gradients tend to vanish due to monotonic decay and are reset after each spike, PMSN overcomes this limitation by introducing oscillatory dynamics governed by the complex-valued  $\bar{\mathcal{T}}$  and unaffected by neuronal resets. As a result, gradient flow is preserved over extended time spans, enabling stable and effective temporal credit assignment.

## 4.3 Experimental Results

In this section, we conduct extensive experiments to validate the effectiveness of the proposed PMSN model, highlighting its strengths in multi-scale spatiotemporal processing, accurate static image classification, accelerated simulation, and efficient computation. Unless otherwise stated, all tested PMSNs have  $n = 5$  compartments to ensure a comparable computational cost to the SOTA parallel spiking neuron model (i.e., 32-receptive-field Sliding Parallel Spiking Neuron (SPSN) [79]).

### 4.3.1 Effective Modeling of Temporal Dependencies across Distinct Timescales

We first compare our PMSN model against other SOTA spiking neuron models and neural architectures on temporal processing tasks involving long-term temporal dependencies. Our experiments are conducted on three widely used benchmarks, including Sequential MNIST (S-MNIST) and Permuted Sequential MNIST (PS-MNIST) datasets with 784 timesteps [189], and SHD spoken digit classification dataset with 250 timesteps [131]. As the results summarized in Table 4.1, PMSN models achieve

**Table 4.1: Comparison of classification accuracy of different neuron models on long-term sequential tasks.**

| Dataset            | Timesteps | Approach                       | Parallelizable | Architecture       | Parameters    | Accuracy               |
|--------------------|-----------|--------------------------------|----------------|--------------------|---------------|------------------------|
| S-MNIST / PS-MNIST | 784       | LIF [21]                       | N              | Feedforward        | 85.1k         | 72.06% / 10.00%        |
|                    |           | GLIF [21]                      | N              | Feedforward        | 87.5k         | 95.27% / N.A.          |
|                    |           | DEXAT [71]                     | N              | Recurrent          | N.A.          | 96.40% / N.A.          |
|                    |           | ALIF [46]                      | N              | Recurrent          | 156.3k        | 98.70% / 94.30%        |
|                    |           | TC-LIF [21]                    | N              | Recurrent          | 155.1k        | 99.20% / 95.36%        |
|                    |           | BHRF [180]                     | N              | Recurrent          | 68.9k         | 99.10% / 95.20%        |
|                    |           | SPSN [79]*                     | Y              | Feedforward        | 52.4k         | 97.20% / 82.84%        |
|                    |           | masked PSN [79]*               | Y              | Feedforward        | 153.7k        | 97.76% / 97.53%        |
|                    |           | PSN [79]*                      | Y              | Feedforward        | 2.5M          | 97.90% / 97.76%        |
|                    |           | Spike-driven Transformer [181] | Y              | Transformer        | 90.0k         | N.A. / 96.21%          |
|                    |           | Binary S4D [182]               | Y              | State-Space Model  | 68.9k         | 99.10% / N.A.          |
|                    |           | GSU [182]                      | Y              | State-Space Model  | 85.5k         | 99.40% / N.A.          |
|                    |           | S5-RF [183]                    | Y              | State-Space Model  | 36.4k         | 98.89% / 95.29%        |
|                    |           | <b>PMSN (Ours)</b>             | <b>Y</b>       | <b>Feedforward</b> | <b>66.3k</b>  | <b>99.40% / 97.16%</b> |
|                    |           |                                |                |                    | <b>156.4k</b> | <b>99.53% / 97.78%</b> |
| SHD                | 250       | Adaptive axonal delay [184]    | N              | Feedforward        | 109.1k        | 92.45%                 |
|                    |           | AdLIF [185]                    | N              | Feedforward        | 38.7k         | 94.19%                 |
|                    |           | TSkip [186]                    | N              | Feedforward        | 1.28M         | 94.71%                 |
|                    |           | LIF [187]                      | N              | Recurrent          | 249.0k        | 84.00%                 |
|                    |           | ALIF [46]                      | N              | Recurrent          | 141.3k        | 84.40%                 |
|                    |           | RadLIF [20]                    | N              | Recurrent          | 3.9M          | 94.62%                 |
|                    |           | TC-LIF [21]                    | N              | Recurrent          | 141.8k        | 88.91%                 |
|                    |           | BHRF [180]                     | N              | Recurrent          | 108.8k        | 92.70%                 |
|                    |           | SPSN [79]*                     | Y              | Feedforward        | 107.1k        | 82.51%                 |
|                    |           | masked PSN [79]*               | Y              | Feedforward        | 122.5k        | 86.00%                 |
|                    |           | PSN [79]*                      | Y              | Feedforward        | 232.5k        | 89.75%                 |
|                    |           | DCLS-Delays [77]               | Y              | Convolution        | 200.0k        | 95.07%                 |
|                    |           | Spikingformer [188]            | Y              | Transformer        | 1.98M         | 82.68%                 |
|                    |           | S5-RF [183]                    | Y              | State-Space Model  | 214.5k        | 91.86%                 |
|                    |           | <b>PMSN (Ours)</b>             | <b>Y</b>       | <b>Feedforward</b> | <b>120.3k</b> | <b>94.25%</b>          |
|                    |           |                                |                |                    | <b>199.3k</b> | <b>95.10%</b>          |

\* Our reproduced results based on publicly available codebases    N.A. These results are not publicly available

the highest accuracies across all spiking neuron models, with fewer or comparable amounts of parameters, demonstrating a superior capacity to capture long-term temporal dependency. Single-compartment models, due to their limited expressiveness, generally perform worse than two-compartment counterparts, including DEXAT [71], TC-LIF [21], RadLIF [20], and Balanced Harmonic Resonate-and-Fire (BHRF) [180]. In addition, PMSN significantly outperforms the recently proposed parallel spiking neuron models PSN, masked PSN, and SPSN [79] that are constrained with a single compartment.

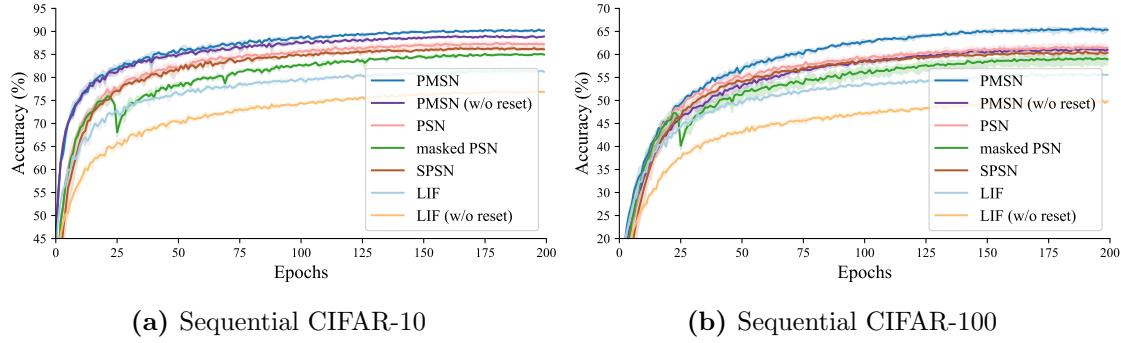


**Figure 4.3:** The learning curves of PMSN models (with and w/o reset), PSN model families, and LIF models (with and w/o reset) on (a) Sequential CIFAR-10 and (b) Sequential CIFAR-100 tasks. The mean (solid lines) and standard deviations (shaded regions) are derived from three independent runs with different random seeds.

**Table 4.2:** Comparison of different models in handling long-term temporal dependencies on Sequential CIFAR-10 and CIFAR-100 datasets.

| Tasks                | Timesteps | PMSN          | PMSN (w/o reset) | PSN    | masked PSN | SPSN   | LIF    | LIF (w/o reset) |
|----------------------|-----------|---------------|------------------|--------|------------|--------|--------|-----------------|
| Sequential CIFAR-10  |           | <b>90.97%</b> | 89.27%           | 88.45% | 85.81%     | 86.70% | 81.50% | 79.50%          |
| Sequential CIFAR-100 | 32        | <b>66.08%</b> | 60.82%           | 62.21% | 60.69%     | 62.11% | 55.45% | 53.33%          |
| No. of Parameters    |           | 0.54M         | 0.54M            | 0.52M  | 0.52M      | 0.51M  | 0.51M  | 0.51M           |
| Sequential CIFAR-10  | 1024      | <b>82.14%</b> | 79.63%           | 55.24% | 57.83%     | 70.23% | 45.07% | 43.30%          |
| No. of Parameters    |           | 0.21M         | 0.21M            | 6.47M  | 0.38M      | 0.18M  | 0.18M  | 0.18M           |

Moreover, PMSN consistently achieves comparable or superior performance to advanced spiking architectures specifically designed for temporal processing, including spike-based transformers (e.g., Spike-driven Transformer [181], Spikingformer [188]), state-space models (e.g., Binary S4D [182], GSU [182], S5-RF [183]), and temporal convolutions (e.g., DCLS-Delays [77]). These findings suggest detailed modeling of structural complexity within neurons as an essential complement to neural architecture in advancing neuromorphic temporal processing.

We further evaluate our model’s ability to establish spatiotemporal and extended long-term temporal dependencies on the more challenging Sequential CIFAR-10 and CIFAR-100 tasks. For Sequential CIFAR-10, we explore two configurations: column-

by-column scanning as per [79] ( $T = 32$ ), which evaluates the model’s capacity in integrating both spatial and temporal information, and pixel-by-pixel scanning ( $T = 1024$ ), which poses a greater challenge to learning long-term dependency. For Sequential CIFAR-100, we use the column configuration. To ensure a fair comparison, we employ the same network architecture for each individual task. As shown in Table 4.2, our PMSN surpasses the SOTA models by at least 2% accuracy, showcasing its superiority in multi-scale temporal processing. As provided in Figure 4.3, the learning curves of the PMSN model exhibit faster and more stable training convergence, aligning well with our theoretical analysis in Section 4.2.4.

Furthermore, we would like to stress the importance of the neuronal reset mechanism that has been neglected by many previous works [79]. As presented in Table 4.2, the accuracy consistently improves for both PMSN and LIF models after incorporating the reset mechanism. This is because the reset mechanism prevents the membrane potential from becoming excessively high, yielding a smoother distribution of membrane potentials across time that can facilitate stable information flow.

### 4.3.2 Superior Accuracy in Static Image Classification

To evaluate our PMSN’s efficacy in non-sequential pattern recognition tasks, we have also conducted a study using the ImageNet-1K [119] dataset, with results detailed in Table 4.3. Our comparison shows that the PMSN achieves the highest accuracy on both ResNet-18 and ResNet-34 backbones. This consistent improvement across both sequential and static benchmarks highlights the effectiveness and scalability of the proposed neuron model. Nevertheless, its performance is marginally lower than that of recent spiking transformer architectures. This result is expected, as transformer-based backbones possess fundamentally different architectural characteristics, offering stronger global representation and scaling capacity than our convolutional backbones. While the primary goal of this work is to advance the temporal modeling capability of

**Table 4.3: Comparison on ImageNet-1K Image Classification Dataset.**

| Approach            | Architecture              | Parameters | Timesteps | Accuracy      |
|---------------------|---------------------------|------------|-----------|---------------|
| Spikformer [190]    | Spiking Transformer-8-384 | 16.8M      | 4         | 70.24%        |
|                     | Spiking Transformer-6-512 | 23.4M      | 4         | 72.46%        |
| MPBN [191]          | ResNet-18                 |            | 4         | 63.14%        |
| InfLoR-SNN [192]    | ResNet-18                 |            | 4         | 64.78%        |
| IF [33]             | SEW ResNet-18             |            | 4         | 63.18%        |
| GLIF [70]           | MS-ResNet-18              | ~12M       | 6         | 68.11%        |
| Attention SNN [158] | MS-ResNet-18              |            | 1         | 63.97%        |
| PSN+TET [79]        | SEW ResNet-18             |            | 4         | 67.63%        |
| <b>PMSN</b>         | <b>SEW ResNet-18</b>      |            | <b>4</b>  | <b>68.77%</b> |
| MPBN [191]          | ResNet-34                 |            | 4         | 64.71%        |
| InfLoR-SNN [192]    | ResNet-34                 |            | 4         | 65.54%        |
| IF [33]             | SEW ResNet-34             |            | 4         | 67.04%        |
| GLIF [70]           | ResNet-34                 | -          | 4         | 67.52%        |
| TET                 | SEW ResNet-34             | ~22M       | 4         | 68.00%        |
| Attention SNN [158] | MS-ResNet-34              |            | 1         | 69.15%        |
| PSN+TET [79]        | SEW ResNet-34             |            | 4         | 70.54%        |
| <b>PMSN</b>         | <b>SEW ResNet-34</b>      |            | <b>4</b>  | <b>70.98%</b> |

SNNs at the neuronal level rather than to pursue architectural innovations, we plan to extend this direction in future work by integrating the computational advantages of PMSN into transformer-based backbones. This will form a hybrid architecture that unifies the global representation capacity of transformers with the multi-scale temporal integration ability of PMSN neurons, thereby enabling further improvements on large-scale vision tasks.

### 4.3.3 Visualization of Multi-compartment Dynamics

Having demonstrated the superior performance of the proposed PMSN models, we next analyze how PMSN integrates multi-scale temporal information through the interactive dynamics of neuronal compartments. To this end, we first perform a

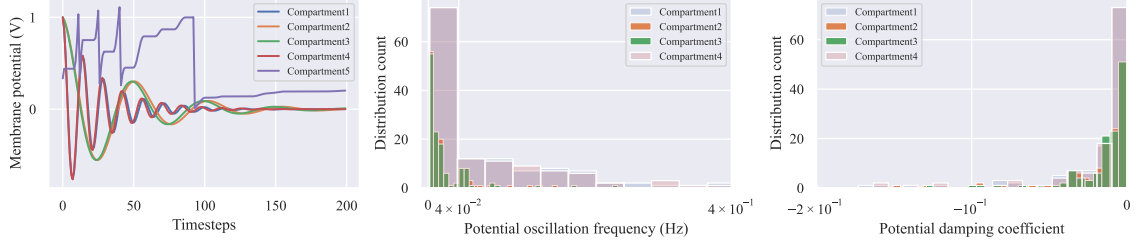


**Figure 4.4: The visualization of PMSN model dynamics ( $n = 5$ ).** **Left:** The impulse response of different compartments within one PMSN neuron, indicating the multi-timescale properties of a single PMSN neuron. Each hidden compartment is characterized by its own dynamic coefficient  $\lambda_i = e^{a+ib}$ , exhibiting damped oscillation patterns after receiving inputs, while the compartment 5 is responsible for spike generation and reset. **Middle:** The distribution of oscillation frequencies  $b/2\pi$ , and **Right:** damping coefficients  $a$  for neurons within the same layer, suggesting different neurons exhibit distinct temporal properties, enabling them to capture temporal dependencies at different timescales.

single-neuron analysis and visualize the dynamics of each compartment in Figure 4.4. We input an impulse at  $T = 0$  and record the trajectory of the membrane potential across compartments as shown in the Left figure. Notably, the interaction within hidden compartments results in damped oscillations of membrane potentials, each with a unique frequency and decay rate, characterized by the dynamic coefficient  $\lambda_i = e^{a+ib}$ . This reflects the system’s multi-scale properties: compartments with higher oscillation frequencies correlate with faster timescales, while lower frequencies indicate slower scales [193]. This compartmental synergy enables the integration of information across various frequency domains and time spans, facilitating multi-scale temporal processing.

Furthermore, we extend this single-neuron analysis to the distribution of oscillation frequency and damping coefficients across neuron populations within one layer. The results provided in the Middle and Right figures reveal diverse compartmental temporal characteristics across the neuron population. The results demonstrate that PMSNs within the same population display heterogeneous information decay rates and varying information integration frequencies. This neuron-wise variability in tem-

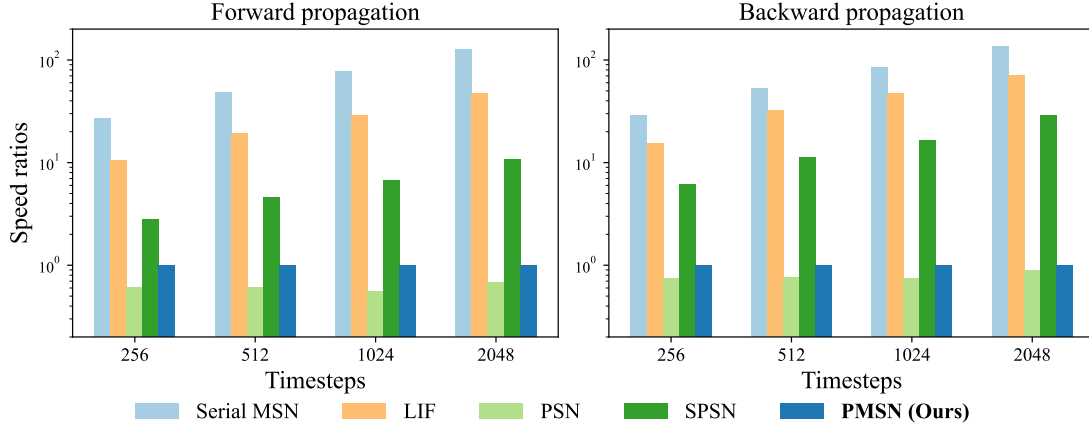


**Figure 4.5: Comparison of simulation speed ratios  $t_i/t_{\text{PMSN}}$  among serial MSN with identical structure, LIF, PSN, SPSN, and PMSN.  $t_i$  represents the recorded runtime per propagation for model  $i$ .**

poral dynamics allows a network constructed from PMSNs to integrate information across a broader spectrum of timescales, significantly enhancing the temporal processing capacity at the network level.

#### 4.3.4 Simulation Acceleration

To quantitatively evaluate the proposed acceleration techniques, we record the actual inference and training time of PMSNs using the GPU-enabled Pytorch library. Besides that, we also compare them against a range of serial and parallel models, including a serial MSN with identical neuronal dynamics of PMSN, LIF, PSN, and SPSN. Figure 4.5 shows the acceleration ratios across diverse sequence lengths under the same single-hidden layer network architecture consisting of 256 neurons. Our parallel scheme shows a substantial speed-up over serial models, achieving training speed-up ratios up to  $134\times$  and  $71\times$  compared to the serial MSN and LIF models, respectively. When compared with SOTA parallel models, our PMSN outperforms SPSN but is slower than PSN. This can be attributed to a larger number of neuronal compartments used (i.e., five in PMSN v.s. one in PSN) and less efficient implementa-

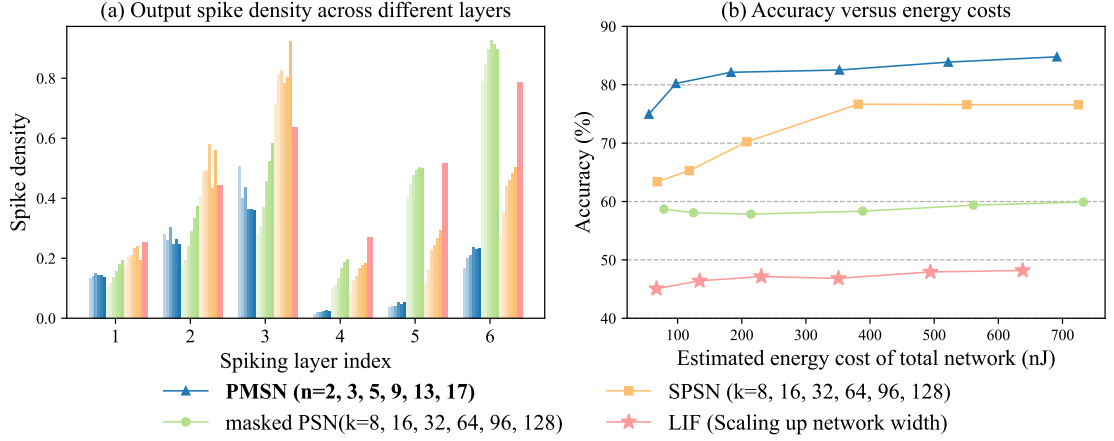


**Figure 4.6: Spike sparsity, accuracy, and empirical energy cost comparison among LIF, PSN families, and our proposed PMSN model across distinct model sizes.** The darkness of bars in panel (a) indicates ascending model sizes.

tion of FFT operations compared to PSN’s matrix multiplication in the current GPU acceleration framework. Nevertheless, this slight slowdown is worthwhile considering the overall effectiveness of our model. Additionally, we observe a positive correlation between the speed-up ratio and sequence length, implying the strength of our model on longer temporal processing.

### 4.3.5 Computational Cost

The PMSN offers an energy-efficient, hardware-friendly solution for practical applications. To shed light on this, we conduct a comparative study to assess the spike sparsity and computational cost of our PMSN, PSN family [79], and LIF models.

The experiments are conducted on the pixel-by-pixel Sequential CIFAR-10 task, with the same network structure for all control models. Note that the total computational cost is affected by both neuronal and synaptic updates. For our PMSN model, scaling up the total number of compartments  $n$  leads to a nearly linear increase in the cost of neuronal updates. Similarly, enlarging the temporal receptive field, denoted as “ $k$ ”, in the  $k$ -order masked PSN and SPSN models has the same effect. For a thorough

**Table 4.4: Comparative analysis of theoretical energy cost among different neuron models with the same network structure.**

| Model              | Neuronal Dynamics                                                                                                              | Theoretical Energy Cost              |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------|--------------------------------------|
| LIF                | $V[t] = \alpha V[t-1] + I[t] - \theta S[t-1]$                                                                                  | $hmtFr_{in}E_{AC} + mtE_{MAC}$       |
| PSN                | $V[t] = \sum_{i=0}^t \mathcal{W}_{t,i}I[i]$                                                                                    | $hmtFr_{in}E_{AC} + mt^2E_{MAC}$     |
| masked PSN         | $V[t] = \sum_{i=t-k+1}^t \mathcal{W}_{t,i}I[i]$                                                                                | $hmtFr_{in}E_{AC} + kmtE_{MAC}$      |
| SPSN               | $V[t] = \sum_{i=t-k+1}^t \mathcal{W}_iI[i]$                                                                                    | $hmtFr_{in}E_{AC} + kmtE_{MAC}$      |
| <b>PMSN (Ours)</b> | $V_h[t] = \mathcal{T}V_h[t-1] + \Phi_c I[t]$ $I_h(t) = \Phi_s V_h(t) + \gamma_n I(t)$ $v_s[t] = v_s[t-1] + I_h[t] - v_r[t-1].$ | $hmtFr_{in}E_{AC} + 8(n-1)mtE_{MAC}$ |

$h$  - input dimension,  $m$  - neuron numbers,  $t$  - simulation time,  $k$  - order of PSN families  
 $Fr_{in}$  - average spike frequency of each presynaptic layer,  $n$  - compartment number of our PMSN

comparison, we scale up the network to derive the LIF neuron’s energy cost curves. In contrast, the cost of synaptic updates primarily depends on the sparsity of output spikes. Notably, as illustrated in Figure 4.6(a), PMSN stands out for having the lowest spike density among all models. To understand how the PMSN model achieves this efficiency, we turn to the multi-compartmental properties that set it apart from other models. The underlying reason can be attributed to two main factors: Firstly, the mutual inhibitory effect resulting from the coupled compartments may contribute to the spike frequency adaptation. Moreover, the enhanced internal dynamics within neuronal compartments may reduce the computation burden on the spike generation, leading to a reduction in spike redundancy.

To further qualify the theoretical and empirical energy consumption of each model, the number of MAC operations and AC operations during one inference are used. The multiplication in synaptic operations between layers employs more efficient AC operations, while the neuronal updates involving analog computation induce more energy-intensive MAC operations. According to the prominent operations in their respective neuronal dynamics, we derive the theoretical cost of each model within

**Table 4.5: Ablation study of compartment number on spatiotemporal integration tasks Sequential CIFAR-10 and CIFAR-100 with  $T = 32$ .**

| Dataset                 | Compartment number $n$ |        |        |        |        |        |
|-------------------------|------------------------|--------|--------|--------|--------|--------|
|                         | 2                      | 3      | 4      | 5      | 9      | 17     |
| Sequential<br>CIFAR-10  | 88.79%                 | 90.49% | 90.75% | 90.97% | 91.05% | 91.43% |
| Sequential<br>CIFAR-100 | 61.84%                 | 65.16% | 65.83% | 66.08% | 66.51% | 66.81% |

a layer as shown in Table 4.4. Following the data collected by [194] on the  $45nm$  CMOS, where  $E_{AC} = 0.9 \text{ pJ}$  and  $E_{MAC} = 4.6 \text{ pJ}$  respectively, we can further estimate empirical energy cost. As indicated by the accuracy-energy curves of different spiking neuron models in Figure 4.6(b), each data point on these curves corresponds to a specific model from Figure 4.6(a). Because the PSN model has a considerably high cost of  $5541 \text{ nJ}$ , it is omitted from this figure. Our PMSN consistently outperforms other models in terms of accuracy when provided with the same amount of energy. Notably, the accuracy of PMSN exhibits rapid improvement as the number of compartments increases from 2 to 5, while the improvement plateaus beyond that point. Therefore, it offers users the flexibility to strike a favorable balance between computational cost and accuracy.

### 4.3.6 Ablation Study

In Figure 4.6(b), we report the accuracy of the PMSN model with different compartment numbers on the pixel-by-pixel Sequential CIFAR-10 task ( $T = 1024$ ). In this subsection, we further evaluate the effectiveness of the number of PMSN compartments on performance using Sequential CIFAR-10 and CIFAR-100 ( $T = 32$ ). As presented in Table 4.5, when the compartment number  $n$  increases from 2 to 5, the accuracy improves rapidly. As  $n$  continues to increase, the accuracy improvement starts

to saturate, but at the cost of a substantial increase in computational consumption. Furthermore, to enable parallelization of the proposed reset mechanism in Eq. (4.16),  $I_h$  is passed through a ReLU function to ensure non-negativity before being injected into the output compartment. To evaluate the effectiveness of this design, we conduct an ablation study to investigate the impact of removing this value clamping in terms of training speed and accuracy. Firstly, according to our analysis, if negative inputs are allowed for the output compartment, the state update of the last compartment can only be done in a serial manner, while other compartments can still be computed in parallel. We have quantified its impact on the simulation speed. The time elapsed (seconds/epoch) for PMSN and last-compartment-serial PMSN are 62 vs 158 on sequential CIFAR-10 (T=32), and 73 vs 5182 on sequential CIFAR-10 (T=1024) tasks. These exploded simulation time underscore the necessity of the proposed parallel solution. Interestingly, we also found that the last-compartment-serial PMSN model, which allows negative input for the last compartment, performs worse than the PMSN model. It exhibits a 3.98% and 4.5% accuracy drop on sequential CIFAR-10 and sequential CIFAR-100 (T=32) tasks, respectively. This performance degradation can be attributed to the negative inputs that shift the distribution of membrane potential  $v_s$  farther away from the firing threshold  $\theta$ , which hinders network convergence [32, 113] and results in poorer classification accuracies than our proposed model.

## 4.4 Discussion and Conclusion

This chapter enhances the structural complexity of spiking neuron models by proposing the PMSN, which captures intricate internal structures of biological neurons to enable multi-scale temporal processing while retaining computational efficiency. Specifically, we propose a generalized multi-compartment neuron model that models each neuron as multiple interacting compartments. Furthermore, we introduce a parallel implementation for this model, enabling accelerated training on GPU-accelerated

ML frameworks. Experimental results demonstrate its exceptional performance in establishing temporal dependencies across various timescales, providing significant training acceleration, and striking a favorable trade-off between accuracy and computational cost. This breakthrough presents abundant opportunities for solving challenging pattern recognition tasks using neuromorphic solutions with high accuracy and computational efficiency.

While the present experiments primarily focus on sequential image and speech tasks, in Chapter 6, we further evaluate PMSN on a diverse set of real-world tasks, such as robotic control, EEG-based motor intention decoding, and sensor-based activity recognition. Across these tasks, PMSN demonstrates competitive performance compared to SOTA spiking neuron models, supporting its applicability in broader real-world scenarios.

Nonetheless, several limitations of the current work should be acknowledged, along with directions for future research. First, implementation on neuromorphic hardware has not yet been conducted. As a result, energy efficiency analysis remains at the theoretical level. In future work, we aim to deploy the PMSN model on event-driven neuromorphic platforms to evaluate its hardware feasibility and empirically measure throughput and power consumption. Second, current evaluations are limited to mid-scale neural architectures. We plan to extend this line of work by integrating the computational advantages of PMSN into transformer-based backbones. This will enable the development of hybrid architectures that combine detailed intra-neuron structural modeling with advanced inter-neuron computational mechanisms, forming a unified framework for high-fidelity temporal modeling in neuromorphic pattern recognition. Third, the current GPU-based machine learning framework does not offer optimal support for the FFT-based computations involved in PMSN. However, this limitation can be addressed through the use of efficient parallel scan algorithms [179] in future work, as demonstrated in recent works [172, 173].

## Chapter 5

# Temporal Structure Encoding Drives Accurate and Robust Sensorimotor Processing in Spiking Neural Networks

This chapter advances spiking neuron models with neural modulation mechanisms by introducing TE-SNNs, which enhance the temporal structure awareness of spiking neurons. Specifically, TE-SNNs integrate a brain-inspired TE module, which encodes task-specific temporal structures and modulates neuronal dynamics, enabling spiking neurons to adapt their excitability to different input patterns. As a result, TE-SNNs excel not only in timing tasks involving interval, duration, and periodicity discrimination, but also achieve significantly improved accuracy and robustness in vision, speech, and text pattern recognition tasks. We also demonstrate practical significance in an autonomous driving scenario, where TE-SNNs produce optimized driving strategies with substantial gains in energy efficiency and robustness, illustrating the potential for deploying advanced spiking neuron models in complex, resource-constrained real-

world environments.

## 5.1 Introduction

The ability to discriminate the temporal structure of events is a fundamental aspect of life for both humans and animals, shaping how we perceive, interact with, and adapt to our environments. From the precise timing required to catch prey or avoid predators in the animal kingdom to the complex rhythms of human speech and body movements, temporal discrimination underpins critical behaviors and decision-making processes.

Recent advances in neuroscience have elucidated specialized neural mechanisms responsible for encoding the temporal structure of events [55, 56, 195, 196, 197]. Broadly, TE is achieved through two distinct forms of neural dynamics: rhythmic and non-rhythmic. Rhythmic TE is widespread throughout the brain. For instance, neural oscillations in the suprachiasmatic nucleus serve as the brain’s central pacemaker, encoding the 24-hour circadian rhythm and regulating daily physiological and behavioral processes [198, 199]. Similarly, oscillatory activity in the inferior olive of the cerebellum plays a critical role in motor learning and the coordination of body movements [200, 201]. In addition to rhythmic TE, studies have also identified non-rhythmic TE mechanisms, which are more task-specific and experience-dependent. For example, hippocampal time cells encode the temporal structure of event sequences through sequential firing patterns that develop with repeated exposure to specific tasks [202, 203]. Similarly, striatal neurons in the basal ganglia encode temporal intervals through ramping activity, characterized by a gradual increase in firing rate, with the pattern of activity varying depending on the task [204, 205].

NC seeks to emulate neurobiological principles to design AI systems with enhanced efficiency and capabilities [10, 14, 15, 16, 157]. Despite the critical role of TE mech-

anisms in temporal discrimination, their application in NC systems remains largely unexplored, significantly limiting the temporal processing capabilities of these systems [206]. To close this gap, we propose TE-SNN, which advances spiking neuron models by incorporating a TE module to encode the essential temporal structures in input signals. Building upon this, we introduce a novel neural modulation mechanism, through which the TE module modulates the temporal dynamics of spiking neurons, thereby augmenting their temporal structure awareness. Using TE-SNN, we demonstrate significant improvements over existing SNNs in discriminating fundamental temporal structures, such as intervals, durations, and periodicity, as well as more complex task-specific temporal structures in visual, speech, and text signals. Furthermore, TE-SNNs show enhanced robustness against a wide range of hardware-related noise due to their improved temporal selectivity. This selectivity allows neurons to produce bursting behaviors in response to learned temporal structures.

The enhanced temporal discrimination capabilities and robustness of TE-SNNs unlock transformative potential for addressing critical challenges in real-world autonomous systems, such as locomotion control and path planning, which demand low-latency, robust, and energy-efficient interactions with dynamic environments. To demonstrate the practical significance of TE-SNNs, we apply the model to an autonomous driving task requiring navigation with exceptional stability and precision. By leveraging the encoded temporal structure of roadmaps learned from past experiences, TE-SNNs generate optimized driving strategies that maximize travel distance within a constrained racing time while achieving significant improvements in energy efficiency and noise robustness. These results not only highlight the superior performance of TE-SNNs in handling real-world temporal complexities but also establish a foundation for next-generation neuromorphic autonomous systems capable of robust and efficient operation in dynamic environments.

The remainder of this chapter is organized as follows: Section 5.2 introduces the TE-SNN models, detailing the design of the TE module and the neural modulation mech-

anism used to regulate spiking neuron dynamics. It also presents a theoretical analysis of how TE-SNNs facilitate effective temporal credit assignment. Section 5.3 provides comprehensive experimental results, demonstrating the effectiveness of TE-SNNs in enhancing temporal structure discrimination, establishing long-term memory, solving temporal processing benchmarks, improving robustness to device-level non-idealities, and enabling real-world deployment in autonomous driving tasks. Finally, Section 5.4 discusses the broader implications, biological inspirations, and potential limitations of the proposed approach.

## 5.2 TE-SNN: A Spiking Neural Network Incorporated with Temporal Structure Encoding

TE represents a fundamental neural mechanism that enables animals to perceive and respond to sensorimotor signals characterized by complex temporal structures. For instance, as illustrated in Figure 5.1(a), when humans dance to music, they effortlessly perceive inter-note intervals, pitch durations, and rhythmic patterns, integrating these temporal elements to coordinate precise body movements. Inspired by this crucial neural mechanism, we propose a TE module that encodes task-specific temporal structures within SNNs. By dynamically modulating neuronal activity, this module enhances the capacity of spiking neurons to discriminate and respond to temporally structured information.

The TE module is formalized as a set of time-varying signals, each capturing a distinct temporal structure. Motivated by biological observations, we design the TE module in two forms: rhythmic and non-rhythmic. As shown in Figure 5.1(b), the rhythmic TE module utilizes cosine functions to model neural oscillatory activities across diverse timescales. In contrast, the non-rhythmic TE module captures task-specific neural activities via learnable, time-dependent functions. These two forms offer distinct

advantages in practice: rhythmic TE is well-suited for processing signals with intrinsic periodicity, such as speech and body movements, while non-rhythmic TE provides greater flexibility and expressiveness for modeling complex and irregular temporal structures. For convenience, we refer to SNNs incorporating a rhythmic TE module as TE-SNN-R, and those with a non-rhythmic TE module as TE-SNN-N.

Moreover, TE signals have been shown to interact with subthreshold membrane dynamics and increase neuronal excitability when external inputs are temporally synchronized with the TE signals [207, 208]. Drawing on this observation, we design the TE module to interact with the membrane potential  $V(t)$  of spiking neurons and dynamically modulate the firing threshold  $\theta(t)$ , as illustrated in Figure 5.1(c). This temporal modulation significantly enhances the temporal structure awareness of SNNs. As shown in Figure 5.1(d), the standard LIF neuron has limited ability to distinguish between spike trains with similar temporal structures. In contrast, Figure 5.1(e) demonstrates that when the TE module is incorporated into the LIF neuron, the resulting TE-LIF neuron excels at discriminating subtle differences in the input spike trains. Specifically, when the temporal dynamics of  $TE(t)$  are aligned with the input, the resulting modulation signal lowers the firing threshold  $\theta(t)$ , which in turn enhances neuronal excitability and facilitates the distinction of inputs with subtle temporal structure differences. In the proposed TE-SNNs, individual neurons are modulated by different TE signals, and the collective network response enables effective discrimination of diverse temporal structures present in the input.

In the following, we first present the detailed formulations of two variants of the TE module, each designed to capture distinct temporal structures. We then detail the mechanism by which TE signals modulate the temporal dynamics of spiking neurons, forming the complete TE-SNN framework.

## 5.2. TE-SNN: A Spiking Neural Network Incorporated with Temporal Structure Encoding

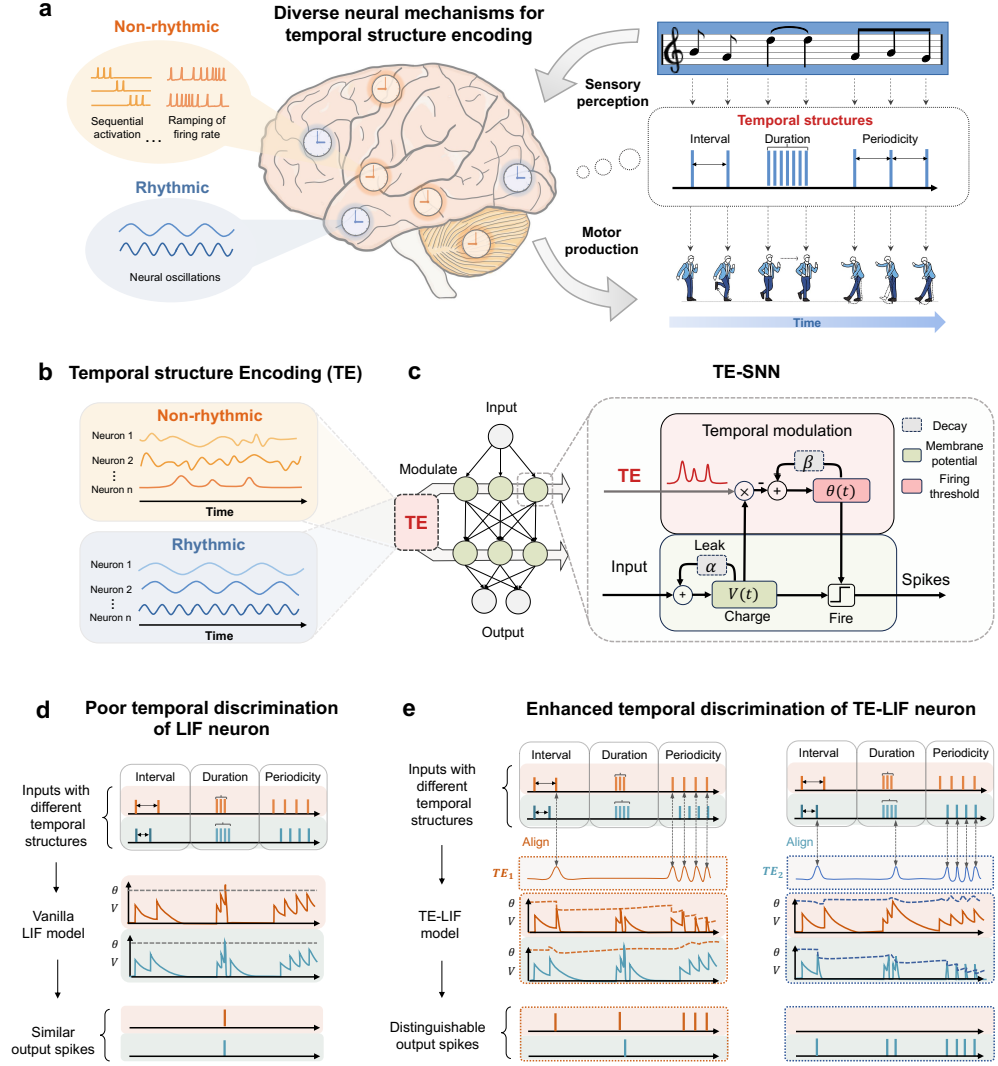


**Figure 5.1: Enhancing temporal processing capacities of SNNs through temporal structure encoding.** (a) The brain excels in perceiving and producing complex temporal structures in sensorimotor signals. This capability largely benefits from its intrinsic neural encoding of temporal structures, manifested through both rhythmic and non-rhythmic neural activities distributed across various brain regions. (b) Inspired by biological mechanisms of temporal structure encoding, we introduce a TE module designed to encode behaviorally significant temporal structures. (c) The TE module modulates the neuronal dynamics of spiking neurons by dynamically adjusting the firing threshold  $\theta(t)$  in response to the temporal structures present in the inputs. (d) Conventional LIF neurons face challenges in distinguishing between different temporal structures in input spike trains. Consequently, crucial information embedded within these temporal structures may be lost. (e) The TE module enhances the temporal structure awareness of SNNs. When the dynamics of the TE module align with the temporal structures present in the input spike train, it triggers burst-firing behavior in the corresponding neurons, indicating the detection of specific temporal structures.

### 5.2.1 Design of the TE Module

Motivated by temporal encoding mechanisms observed in biological neural systems, the TE module is designed to generate a set of time-varying modulation signals that explicitly encode temporal structures. Specifically, the dynamics of the TE module are formulated as an  $n$ -channel signal vector:

$$\mathbf{TE}[t] = \left[ TE_1[t], \dots, TE_n[t] \right], \quad (5.1)$$

where  $n$  is the number of channels, and each  $TE_i[t]$  ( $i = 1, \dots, n$ ) represents the temporal signal assigned to the  $i^{th}$  channel. These channels exhibit distinct temporal dynamics, collectively enabling the encoding of diverse temporal structures across multiple timescales. Throughout the paper, bold symbols (e.g.,  $\mathbf{TE}[t] \in \mathbb{R}^n$ ) indicate multichannel vectors, while regular symbols (e.g.,  $TE_i[t] \in \mathbb{R}$ ) refer to individual channel components. To model different types of temporal structures, we introduce two variants of TE modules: rhythmic TE and non-rhythmic TE.

The rhythmic TE module is inspired by oscillatory neural activity with different frequencies observed in cortical and hippocampal circuits. Each channel in this module functions as a cosine-based oscillator with a unique frequency to encode periodic temporal structures across multiple timescales:

$$\mathbf{TE}_R[t] = \left[ A \cos(\omega_1 t + \phi), \dots, A \cos(\omega_n t + \phi) \right], \quad (5.2)$$

where  $\omega_i$  ( $i = 1, \dots, n$ ) denotes the frequency of the  $i^{th}$  oscillator,  $A$  is the amplitude (fixed to 0.1), and  $\phi$  is the phase offset (set to 0). To ensure the TE module can capture temporal structures across a wide range of timescales, the oscillation frequencies  $\omega_i$  are distributed across a broad spectrum. This is achieved by scaling  $\omega_i$  as:

$$\omega_i = w_i \cdot \Delta t_i, \quad (5.3)$$

where  $w_i$  is a learnable base frequency.  $\Delta t_i$  is a scaling factor sampled logarithmically from  $1 \times 10^{-3}$  to  $1 \times 10^{-1}$ , covering both fine-grained and long-term temporal structures.

In contrast, to capture task-specific and irregular temporal structures, the non-rhythmic TE module models each channel as a learnable function:

$$\mathbf{TE}_N[t] = \left[ f_1(t), \dots, f_n(t) \right]. \quad (5.4)$$

Each function  $f_i(t)$  ( $i = 1, \dots, n$ ) defines the temporal dynamics for the  $i^{th}$  channel, with its output at every timestep  $t$  treated as an independent learnable parameter.

For all experiments, the temporal functions  $f_i(t)$  and base frequencies  $w_i$  are learned jointly with network parameters. Their initial values are set as:

$$f_i(t) \sim \mathcal{N}(0.01, 0.01), \quad w_i \sim \mathcal{U}[2\pi, 4\pi), \quad (5.5)$$

where  $\mathcal{N}$  and  $\mathcal{U}$  denote the normal and uniform distributions, respectively.

### 5.2.2 Integration of TE Modulation Mechanism into Spiking Neuron Models

To leverage the temporal structures encoded in the TE module, we propose a neural modulation strategy where TE signals dynamically modulate neuronal excitability. This allows neurons to selectively respond to inputs aligned with these encoded temporal structure, thereby giving rise to TE-SNNs.

Here, we illustrate this modulation strategy using the LIF neuron model, a widely used spiking neuron model known for its simplicity and analytical tractability [51]. For clarity, we omit the channel index  $i$  and present the standard LIF dynamics as:

$$\begin{cases} V[t] = \alpha \cdot V[t-1] \cdot (1 - S[t-1]) + I[t], \\ S[t] = H(V[t] - \theta[t]), \end{cases} \quad (5.6)$$

where  $V[t]$ ,  $S[t]$ , and  $\theta[t]$  represent the membrane potential, output spike, and firing threshold, respectively.  $H(\cdot)$  is the Heaviside function, and  $\alpha$  is the membrane potential decay factor. In the charging phase, the membrane potential  $V[t]$  integrates

synaptic input current while gradually decaying over time. When  $V[t]$  surpasses the firing threshold  $\theta[t]$ , the neuron emits a spike, after which  $V[t]$  is reset to zero.

TE signals modulate the neuronal excitability by adjusting the firing threshold  $\theta[t]$  based on the product of the TE signal and  $V[t]$ , which can be formalized as:

$$\theta[t] = \beta \cdot (\theta[t-1] - \theta_0) - TE[t] \cdot V[t-1] + \theta_0, \quad (5.7)$$

where  $\theta_0$  represents the initial firing threshold, and  $\beta$  is a slow-decay factor governing how long the modulation persists. The neuronal firing threshold  $\theta[t]$  temporarily adapts to the alignment between membrane potential  $V[t]$  and the temporal structure encoding  $TE[t]$ , thereby increasing neuronal excitability. The threshold adaptation mechanism encompasses three significant advantages: Firstly, the regulated firing threshold can effectively suppress excessive firing, thereby improving the efficiency of neural signal processing [209]. Secondly, the slow decay of  $\beta$  and the structured non-monotonicity of  $TE[t]$  help maintain a stable gradient path, facilitating long-term temporal credit assignment (see the visualization in Figure 5.3(d)). Most significantly, it fosters the emergence of neuron-wise selectivity for temporal structures. Once the temporal dynamics of  $V[t]$  align with  $TE[t]$ , the neuron will undergo a consistent reduction in the firing threshold  $\theta[t]$ , ultimately resulting in a burst-firing behavior that signifies the recognition of specific temporal structures (as shown in Figures 5.5(b, c)). Therefore, this modulation mechanism enables TE-modulated neurons to overcome the limitations of conventional LIF neurons in discriminating diverse temporal structures.

Building on the temporal modulation mechanism, the TE module can be seamlessly integrated into existing SNN architectures. Specifically, the layer-wise dynamics of a TE-SNN are defined as:

$$\begin{cases} \boldsymbol{\theta}^l[t] = \beta \cdot (\boldsymbol{\theta}^l[t-1] - \boldsymbol{\theta}_0) - \mathbf{TE}[t] \odot \mathbf{V}^l[t-1] + \boldsymbol{\theta}_0, \\ \mathbf{V}^l[t] = \alpha \cdot \mathbf{V}^l[t-1] \odot (1 - \mathbf{S}^l[t-1]) + \mathbf{I}^l[t], \\ \mathbf{S}^l[t] = H(\mathbf{V}^l[t] - \boldsymbol{\theta}^l[t]), \end{cases} \quad (5.8)$$

where bold symbols  $\mathbf{V}^l[t]$ ,  $\boldsymbol{\theta}^l[t]$ ,  $\mathbf{S}^l[t]$  represent state vectors of all neurons in layer  $l$  and  $\odot$  denotes the Hadamard product.

The input current to neurons at layer  $l$  is given by:

$$\mathbf{I}^l[t] = \mathcal{W}^{l-1} \mathbf{S}^{l-1}[t] + \mathcal{R}^l \mathbf{S}^l[t], \quad (5.9)$$

where  $\mathbf{S}^{l-1}[t]$  and  $\mathbf{S}^l[t]$  denote the presynaptic spikes from layer  $l-1$  and the output spikes of layer  $l$ , respectively.  $\mathcal{W}^{l-1}$  is the feedforward synaptic weight from layer  $l-1$  to  $l$ , and  $\mathcal{R}^l$  is the recurrent synaptic weight within layer  $l$ . We term the TE-SNN network architecture with only feedforward connections as TE-SFNN and with both feedforward and recurrent connections as TE-SRNN. It is also worth noting that the TE-SNN has been designed to employ a shared TE module across all layers to reduce the memory cost.

### 5.2.3 Effective Temporal Credit Assignment in TE-SNNs

TE-SNNs effectively mitigate the gradient vanishing problem in conventional SNNs, which commonly arises when training on long sequences using the BPTT algorithm with surrogate gradients [24, 25]. Specifically, as demonstrated in Figure 5.3(d), TE-SNNs propagate gradients to much earlier timesteps compared to existing SNN models, thereby ensuring the establishment of long-term temporal dependencies. In this subsection, we provide a theoretical proof of this point.

The update rule of network parameters in TE-SFNNs can be expressed as follows:

$$\Delta \mathcal{W}^l \propto \frac{\partial \mathcal{L}}{\partial \mathcal{W}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]} \mathbf{S}^{l-1}[t], \quad \Delta \mathbf{b}^l \propto \frac{\partial \mathcal{L}}{\partial \mathbf{b}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]}, \quad (5.10)$$

where  $\mathcal{L}$  is the global loss function.  $\mathcal{W}^l$  and  $\mathbf{b}^l$  denote the synaptic weight and bias of layer  $l$ , respectively. Their updates are determined by the partial derivative  $\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]}$ ,

which can be further decomposed using the chain rule as:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} \frac{\partial \mathbf{S}^l[t]}{\partial \mathbf{V}^l[t]} + \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t+1]} \frac{\partial \mathbf{V}^l[t+1]}{\partial \mathbf{V}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t+1]} \frac{\partial \boldsymbol{\theta}^l[t+1]}{\partial \mathbf{V}^l[t]} \\
&= \frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} g'[t] + \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t+1]} \alpha(1 - \mathbf{S}^l[t]) - \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t+1]} \mathbf{T}\mathbf{E}[t+1], \\
\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t+1]} \frac{\partial \mathbf{V}^l[t+1]}{\partial \mathbf{S}^l[t]} + \frac{\partial \mathcal{L}}{\partial \mathbf{V}^{l+1}[t]} \frac{\partial \mathbf{V}^{l+1}[t]}{\partial \mathbf{S}^l[t]} = -\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t+1]} \alpha \mathbf{V}^l[t] + \frac{\partial \mathcal{L}}{\partial \mathbf{V}^{l+1}[t]} \mathcal{W}^l, \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t]} &= \frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} \frac{\partial \mathbf{S}^l[t]}{\partial \boldsymbol{\theta}^l[t]} + \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t+1]} \frac{\partial \boldsymbol{\theta}^l[t+1]}{\partial \boldsymbol{\theta}^l[t]} = -\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} g'[t] + \frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t+1]} \beta,
\end{aligned} \tag{5.11}$$

where  $g'[t] \approx \frac{\partial \mathbf{S}^l[t]}{\partial \mathbf{V}^l[t]} = -\frac{\partial \mathbf{S}^l[t]}{\partial \boldsymbol{\theta}^l[t]}$  refers to the surrogate gradient function, which approximates the non-differentiable derivative of the spike function  $\mathbf{S}^l[t] = H(\mathbf{V}^l[t] - \boldsymbol{\theta}^l[t])$  during backpropagation. In particular, we adopt a rectangle function [24] that is defined as

$$g'[t] = \begin{cases} 1, & \text{if } |\mathbf{V}^l[t] - \boldsymbol{\theta}^l[t]| < \Gamma, \\ 0, & \text{otherwise,} \end{cases} \tag{5.12}$$

where  $\Gamma$  is a hyperparameter that determines the permissible range for gradient propagation. Subsequently, by transforming Eq. (5.11) into a cumulative form using recursive accumulation, we can derive:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^l[t]} = \sum_{i=t}^T -\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[i]} g'[t] \beta^{i-t} = \sum_{i=t}^T -\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[i]} \frac{\partial \mathbf{S}^l[i]}{\partial \mathbf{V}^l[i]} \beta^{i-t} = \sum_{i=t}^T -\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[i]} \beta^{i-t}. \tag{5.13}$$

By substituting Eq. (5.13) into Eq. (5.11), we can derive the gradient received by each neuron at each timestep as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]} &= \underbrace{\frac{\partial \mathcal{L}}{\partial \mathbf{S}^l[t]} g'[t]}_{\text{LIF (Spatial)}} + \underbrace{\frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t+1]} \alpha(1 - \mathbf{S}^l[t])}_{\text{LIF (Temporal)}} + \underbrace{\sum_{i=t+1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[i]} \beta^{i-t-1} \mathbf{T}\mathbf{E}[t+1]}_{\text{TE Modulation}}.
\end{aligned} \tag{5.14}$$

The first two terms in this formulation have an identical format to spatiotemporal gradient propagation in the vanilla LIF model formulated in Eq. (5.6), which is prone to diminishing to zero for earlier timesteps, thereby hindering effective learning of long-term temporal dependencies [21]. Specifically, the first term represents the spatial credit assignment between consecutive layers, which encounters the dead neuron

problem when the surrogate gradient value  $g'[t]$  equals zero. The second term corresponds to the temporal credit assignment that decays monotonously at a rate of  $\alpha$  and resets after each spike generation. Consequently, the gradient information cannot be effectively propagated over a long time span. Notably, our proposed TE-SNN can effectively address this challenge through the introduction of the third term, which establishes a gradient highway among all timesteps. In contrast to the LIF model, where the gradient contribution from earlier timesteps naturally decays over time, the TE module enables flexible adjustment of the gradient assigned to different timesteps by scaling its value with a tunable factor  $\mathbf{TE}[t+1]$ . This mechanism allows TE-SNNs to effectively assign larger gradients to early but critical timesteps and facilitate the establishment of long-range dependencies required for complex temporal processing.

## 5.3 Experimental Results

In this section, we first validate the temporal structure discrimination and long-term temporal dependency modeling capacities of TE-SNNs. Next, we compare the performance of TE-SNNs with both spiking and non-spiking baselines across diverse real-world temporal processing benchmarks. Furthermore, we investigate the robustness of TE-SNNs against device non-idealities and noise perturbations. Finally, we demonstrate the real-world applicability of TE-SNNs through their deployment in an autonomous driving scenario.

### 5.3.1 TE-SNNs Enhance Temporal Structure Discrimination

The ability to discriminate temporal structures, including intervals, durations, and temporal synchronization, is essential for effectively perceiving and responding within dynamic environments. Precise interval discrimination allows organisms to perceive and differentiate the time elapsed between two discrete events, such as the intervals

between auditory beeps or visual flashes. In contrast, duration encoding pertains to the perception of how long a single event or stimulus persists, contributing to the subjective experience of time. Temporal synchronization, on the other hand, facilitates the coordination of neural activity with external rhythms, thereby playing a significant role in motor coordination, sensory integration, and communication.

As illustrated in Figures 5.2(a-c), we conducted three experiments based on well-established experimental paradigms in neuroscience to demonstrate the enhanced temporal structure discrimination capability of TE-SNNs in these fundamental timing tasks [210, 211] (see detailed task formulations in Section 5.3.6). We compare the performance of our TE-SRNN against three SOTA SNN models optimized for temporal processing: (1) a feedforward SNN enhanced with axonal delays to improve temporal alignment [212], (2) a recurrent SNN (SRNN) [35] that leverages dynamic network properties, (3) an SRNN with an adaptive threshold mechanism (ASRNN) to strengthen working memory [46]. As shown in Figures 5.2(d-f), TE-SRNNs exhibit significantly faster and more stable training convergence across all tasks, attributable to their superior temporal structure modeling. Furthermore, due to their inherent capacity for encoding rhythmic temporal structures, TE-SRNN-R outperforms TE-SRNN-N in the synchronization task.

To gain insights into how TE-SNNs effectively solve these timing tasks, we visualize neuronal activity across different layers, as illustrated in Figures 5.2(a-c). In both the Interval and Duration Discrimination tasks, TE-SNNs progressively integrate evidence over time to reach the output decision. This process parallels the evidence accumulation phenomenon observed in psychological decision-making experiments [213, 214]. Another notable finding is that in the Synchronization task, once the network receives periodic inputs, the spiking neurons in the two hidden layers gradually transition into an oscillatory state, with the oscillation period aligning with the ISI of the input layer. This emergent neural oscillation corresponds with the neural synchronization phenomenon widely observed in biological systems [215, 216].

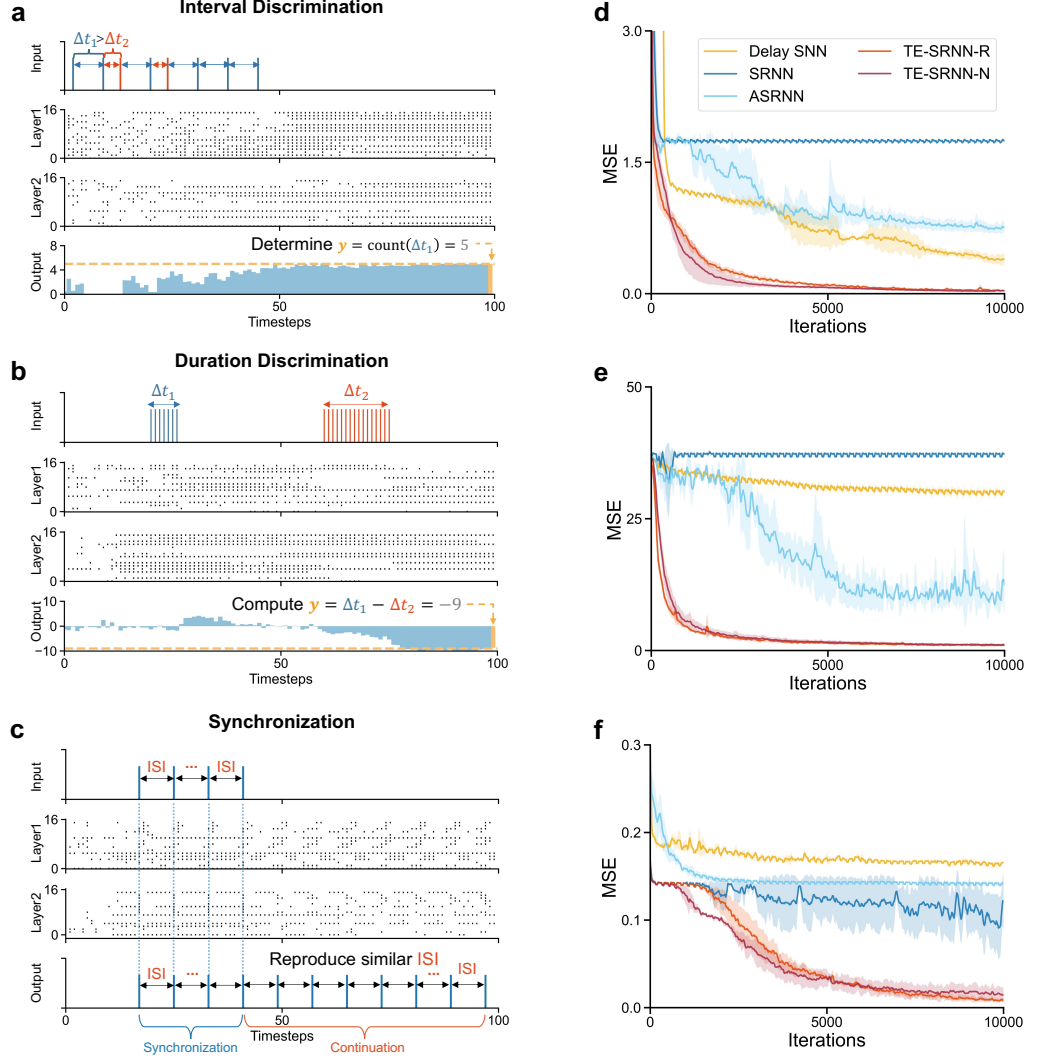


**Figure 5.2: Evaluation of the temporal discrimination capability across three timing tasks.** (a) **Interval Discrimination.** Networks are tasked with discriminating between two distinct intervals and outputting the count of longer intervals at the final timestep. (b) **Duration Discrimination.** Networks are tasked with identifying the durations of two spike-bursting events and outputting the difference in their durations. (c) **Synchronization.** This task consists of two phases: during the synchronization phase, networks are expected to produce outputs synchronized with the rhythmic input presented at a constant ISI. In the continuation phase, networks continue to generate synchronized outputs at the specified ISI in the absence of external stimuli. (d-f) Comparison of the learning curves of different models across three random runs, where the shaded area represents standard deviation and the solid line indicates the average curve. MSE is adopted as the evaluation metric. All models evaluated in the same task have comparable parameter numbers.

### 5.3.2 TE-SNNs Enhance Temporal Processing Involving Long-term Temporal Dependencies

Real-world temporal processing tasks often span longer durations and involve more complex temporal structures than the three timing tasks evaluated earlier. In this subsection, we demonstrate the enhanced capability of TE-SNNs in addressing these challenging temporal processing tasks, particularly those involving long-term temporal dependencies. To this end, we first design a Delayed Recall task, which requires a model to store the input sequence presented at the beginning and accurately recall it after a delay period, as illustrated in Figure 5.3(a). This task requires a model to establish long-term temporal dependencies between the output and the distant input. As shown in Figure 5.3(b), TE-SNNs exhibit a significant advantage over other baseline temporal processing models, achieving the lowest loss values at a sequence length of  $T = 100$ .

To further evaluate the memory capacity of different models, we compare their recall accuracies at sequence length  $T$  of 50, 100, 200, and 400. The recall accuracy is measured as the percentage of correctly recalled numbers, with a baseline accuracy of 12.5%, corresponding to random guessing among eight possible numbers. As shown in Figure 5.3(c), even for the shortest sequence length ( $T = 50$ ), the best-performing baseline model, ASRNN, achieves only 63% accuracy. As the sequence length  $T$  increases, its performance declines sharply, indicating that these models struggle to retain memory over extended delays. In contrast, TE-SRNN-N maintains 100% accuracy across all sequence lengths, demonstrating exceptional memory capacity. The introduced TE module exhibits notable functional parallels to hippocampal time cells, facilitating accurate recall of past events across extended timescales [202, 217].

To understand how TE-SNNs effectively establish long-term temporal dependencies, we visualize the error gradients calculated at different timesteps of the last hidden layer in Figure 5.3(d). Given that key information appears within the first 10

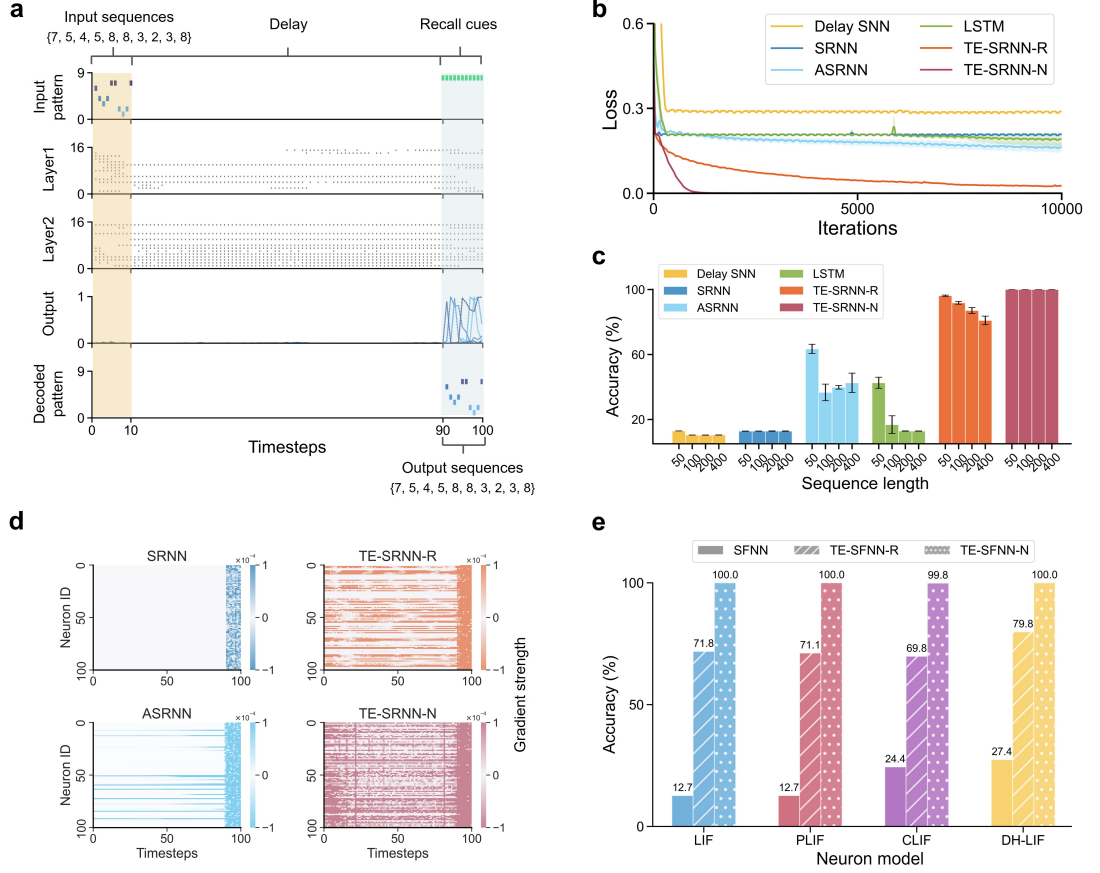


**Figure 5.3: Evaluation of long-term temporal dependency modeling capability using the Delayed Recall task.** (a) In this task, models are required to store the input sequence presented during the first 10 timesteps and reproduce it during the last 10 timesteps after a prolonged delay without any inputs. (b) Learning curves for different models with a sequence length of  $T = 100$ . (c) Accuracy plots across varying sequence lengths, with the mean accuracy and standard deviation (error bar) calculated over three independent runs using different random seeds. (d) Visualization of the error gradients calculated for different timesteps. (e) Accuracy plots for various spiking neuron models with  $T = 400$  illustrate the generalizability of the TE module in enhancing temporal processing across different neuron types.

timesteps, an effective learning algorithm must propagate error gradients to these early timesteps. TE-SNNs successfully achieve this, with substantial gradients observed across nearly all hidden neurons. In contrast, SRNN and ASRNN models suffer from the gradient vanishing problem, particularly the SRNN model. This observa-

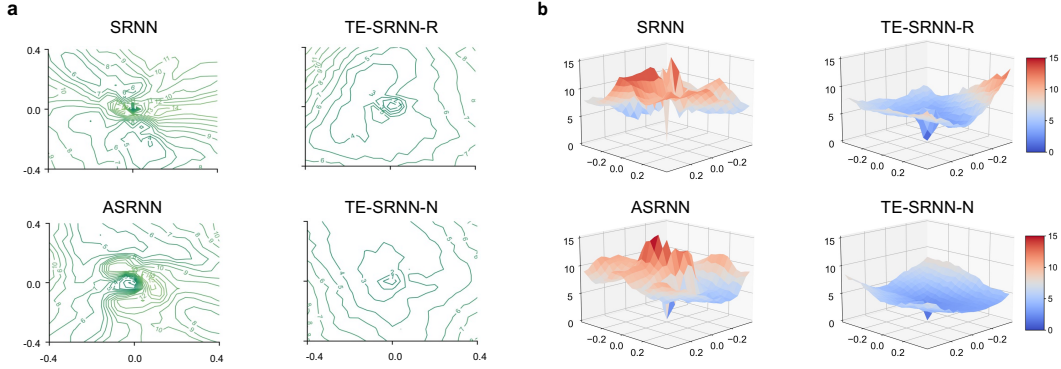


**Figure 5.4: Loss landscape visualization.** Loss landscapes of SRNN, ASRNN, TE-SRNN-R, and TE-SRNN-N on the S-MNIST task are visualized using **(a)** two-dimensional contour plots and **(b)** three-dimensional surface plots. These plots depict the error surface over the parameter space, where x- and y-axes represent different shifting directions in the parameter space, while contour values (2D) or z-axis (3D) represent the loss value. A smoother loss landscape is indicative of easier optimization, characterized by fewer local minima, more stable gradients, faster convergence during training, and improved generalization to unseen data [218].

tion empirically validates our theoretical analysis in Section 5.2.3, where we demonstrate that temporal modulation via the TE module facilitates long-term temporal credit assignment by establishing temporal shortcuts during gradient backpropagation, effectively mitigating the vanishing and exploding gradient problems commonly encountered in long sequence learning.

While the TE module’s effectiveness has been demonstrated with LIF neurons, it is also applicable to other spiking neuron models. To demonstrate this, we incorporate the TE module into PLIF [68], CLIF [75], and DH-LIF [47], and evaluate their performance under the challenging scenario of  $T = 400$  (See Section 5.3.6 for implementation details). As shown in Figure 5.3(e), the inclusion of both rhythmic and non-rhythmic TE modules significantly enhances temporal processing across all models, underscoring the TE module’s broad applicability in various neuromorphic computing systems.

### 5.3. Experimental Results

**Table 5.1:** Comparison of model performance on real-world temporal processing benchmarks.

| Dataset  | Model                   | SNN        | Architecture       | Parameters   | Accuracy↑ / Perplexity↓ |
|----------|-------------------------|------------|--------------------|--------------|-------------------------|
| PS-MNIST | SFNN [206]              | Yes        | Feedforward        | 0.09M        | 57.45%                  |
|          | <b>TE-SFNN-R</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.08M</b> | <b>97.23%</b>           |
|          | <b>TE-SFNN-N</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.09M</b> | <b>98.24%</b>           |
|          | LSTM [219]              | No         | Recurrent          | 0.07M        | 88.00%                  |
|          | ASRNN [46]              | Yes        | Recurrent          | 0.16M        | 94.30%                  |
|          | <b>TE-SRNN-R</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.15M</b> | <b>97.88%</b>           |
|          | <b>TE-SRNN-N</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.15M</b> | <b>98.33%</b>           |
| S-MNIST  | SFNN [21]               | Yes        | Feedforward        | 0.09M        | 72.06%                  |
|          | <b>TE-SFNN-R</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.08M</b> | <b>98.40%</b>           |
|          | <b>TE-SFNN-N</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.09M</b> | <b>98.45%</b>           |
|          | LSTM [219]              | No         | Recurrent          | 0.07M        | 98.20%                  |
|          | LSNN [35]               | Yes        | Recurrent          | 0.07M        | 96.40%                  |
|          | AHP [73]                | Yes        | Recurrent          | 0.07M        | 96.00%                  |
|          | ASRNN [46]              | Yes        | Recurrent          | 0.16M        | 98.70%                  |
|          | <b>TE-SRNN-R</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.15M</b> | <b>98.87%</b>           |
|          | <b>TE-SRNN-N</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.15M</b> | <b>99.00%</b>           |
| SHD      | SFNN [131]              | Yes        | Feedforward        | 0.11M        | 48.60%                  |
|          | SRNN w/ data aug. [131] | Yes        | Recurrent          | 1.79M        | 83.20%                  |
|          | <b>TE-SFNN-R</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.04M</b> | <b>88.69%</b>           |
|          | <b>TE-SFNN-N</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>0.04M</b> | <b>89.62%</b>           |
|          | Bi-LSTM [46]            | No         | Recurrent          | 1.31M        | 87.20%                  |
|          | SRNN [187]              | Yes        | Recurrent          | 0.25M        | 84.00%                  |
|          | ASRNN [46]              | Yes        | Recurrent          | 0.14M        | 87.80%                  |
|          | <b>TE-SRNN-R</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.07M</b> | <b>91.03%</b>           |
|          | <b>TE-SRNN-N</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.07M</b> | <b>89.80%</b>           |
| GSC      | SRNN [220]              | Yes        | Recurrent          | 0.04M        | 86.70%                  |
|          | LSNN [72]               | Yes        | Recurrent          | 4.31M        | 89.04%                  |
|          | LSNN with SFA [72]      | Yes        | Recurrent          | 4.31M        | 91.21%                  |
|          | ASRNN [46]              | Yes        | Recurrent          | 0.22M        | 92.10%                  |
|          | <b>TE-SRNN-R</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.22M</b> | <b>93.56%</b>           |
|          | <b>TE-SRNN-N</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>0.17M</b> | <b>94.34%</b>           |
| PTB      | SFNN [206]              | Yes        | Feedforward        | 4.67M        | 129.96                  |
|          | <b>TE-SFNN-R</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>4.67M</b> | <b>127.68</b>           |
|          | <b>TE-SFNN-N</b>        | <b>Yes</b> | <b>Feedforward</b> | <b>4.74M</b> | <b>112.35</b>           |
|          | LSTM [219]              | No         | Recurrent          | 13.47M       | 106.87                  |
|          | SRNN [206]              | Yes        | Recurrent          | 5.98M        | 111.96                  |
|          | <b>TE-SRNN-R</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>5.98M</b> | <b>109.17</b>           |
|          | <b>TE-SRNN-N</b>        | <b>Yes</b> | <b>Recurrent</b>   | <b>6.05M</b> | <b>99.93</b>            |

Finally, we conduct extensive experiments on real-world temporal processing benchmarks involving long-term temporal dependencies, including pixel-level image clas-

sification on the S-MNIST and PS-MNIST datasets [189], speech classification on the SHD [131] and GSC [129] datasets, and language modeling on the Penn Treebank (PTB) [221] dataset. As shown in Table 5.1, both TE-SFNN and TE-SRNN demonstrate exceptional performance, surpassing advanced SNN and non-spiking Long Short-Term Memory (LSTM) models with fewer or comparable model parameters. The effectiveness of TE-SNNs in these challenging temporal processing tasks is further evidenced by their smoother loss landscapes compared to other SNN models, as demonstrated in Figure 5.4. These smooth loss landscapes enable rapid and stable convergence for TE-SNNs, while also promoting effective generalization to unseen data [218].

### 5.3.3 TE-SNNs Improve Robustness against Device Non-idealities and Noise Perturbations

The burst-firing activity of biological neurons plays a pivotal role in enhancing the robustness of neural computations, enabling neurons to reliably transmit information despite fluctuating signals and noise [222]. This characteristic is particularly advantageous for analog neuromorphic computing substrates, which frequently encounter similar non-ideality issues. Here, we demonstrate that TE-SNNs exhibit robust performance against device non-idealities and noise perturbations by leveraging the burst-firing behavior facilitated by the temporal structure encoding.

As illustrated in Figure 5.5(a), we evaluate three primary sources of noise commonly encountered in neuromorphic systems: (1) synaptic noise, (2) neuronal noise, and (3) background noise. Each noise source is modeled using two distinct noise types, each applied at five progressively increasing levels (Levels 1–5) to systematically assess model robustness. Synaptic noise includes quantization noise and device mismatch noise, which arise from limited precision and variability in synaptic weights [223]. Neuronal computation on neuromorphic chips is often affected by thermal noise and

spike silencing noise, which disrupt information transmission among neurons. Background noise, originating from sensor inputs, includes pepper noise and spike jittering noise. Detailed experimental settings are provided in Section 5.3.6.

Our evaluation results reveal a stark contrast in robustness between TE-SNNs and conventional SNN models, such as SRNN and ASRNN. While the accuracy of SRNN and ASRNN degrades rapidly across all noise types, TE-SNNs exhibit remarkable robustness, maintaining stable performance even under severe noise conditions. For example, TE-SRNN-N demonstrates exceptional robustness to weight quantization, achieving 92% accuracy even at 3-bit precision (Level 5), whereas SRNN and ASRNN drop to near-random guessing levels. Similar trends are consistently observed across all other noise types, underscoring the superior noise robustness of TE-SNNs in handling a wide range of real-world noise conditions.

To further illustrate how the TE module enhances noise robustness, we compare the output neuron responses under both clean and pepper-noise-corrupted inputs in Figures 5.5(b, c). In this experiment, images from the MNIST dataset are presented to the TE-SNNs pixel-by-pixel following a raster scan. For clean inputs, the temporal structure stored in  $TE_i$ , corresponding to class  $i$ , aligns closely with the membrane potential  $V_i$  of the target neuron. This suggests that  $TE_i$  has effectively learned to encode the stereotypical temporal structures of digit  $i$  during training. By detecting the alignment between  $TE_i$  and  $V_i$ , the target neuron identifies the presence of digit  $i$  in the input sequences, leading to a consistent reduction in the firing threshold  $\theta$  and a burst-firing response. This burst-firing behavior indicates enhanced neuronal selectivity to class-dependent temporal structures, enabling TE-SNNs to effectively discriminate between different digits. Remarkably, for noisy inputs,  $V_i$  deviates significantly from the clean scenario, which would typically result in a substantial decrease in classification accuracy. However, facilitated by the TE module, TE-SNNs can reliably detect the alignment between  $TE_i$  and  $V_i$  despite noise perturbations.

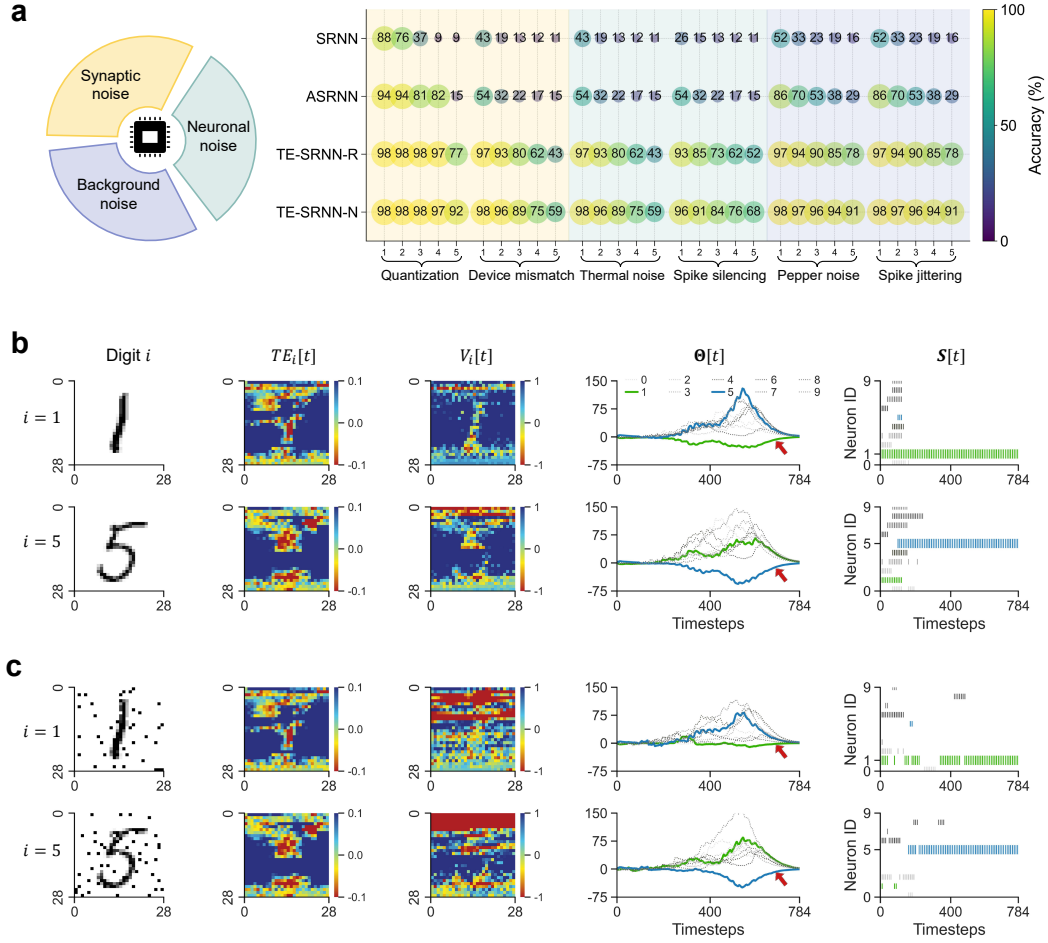


**Figure 5.5: Evaluation of robustness against device non-idealities and noise perturbations.** (a) Performance comparison under various types and levels of noise. The diameter of each bubble is proportional to the test accuracy, as indicated by the values displayed within the bubbles. (b, c) To elucidate the exceptional robustness of TE-SNNs to noise perturbations, we visualize the dynamics of output neurons under both clean and noise-corrupted inputs using the S-MNIST task. The input image of digit  $i$  is fed into the model pixel-by-pixel via a raster scan. Neuronal states, including the temporal structure encoding  $TE_i$  and membrane potential  $V_i$  of the target neuron, as well as firing thresholds  $\theta$  and output spike trains  $S$  across all output neurons, are recorded over time. For clarity, the recorded states are reshaped to match the dimensions of the input images. Under clean conditions, as illustrated in (b), the trajectories of  $V_i$  synchronize with  $TE_i$ , leading to a significant reduction in the firing threshold of the target neuron (indicated by the red arrow) and triggering a burst-firing response, while other output neurons remain less active. This demonstrates the enhanced selectivity of TE-SNNs for class-specific temporal structures. Equipped with this selectivity, TE-SNNs exhibit strong robustness to the background noise, as illustrated in (c).

### 5.3.4 Application of TE-SNNs in Autonomous Driving

The enhanced temporal processing capability and robustness of TE-SNNs are essential for real-world autonomous systems, which must operate reliably in dynamic and unpredictable environments. To demonstrate the broad applicability of TE-SNNs in such scenarios, we design an autonomous driving task in which a four-wheeled vehicle with Ackermann steering learns to navigate a Formula One track, as illustrated in Figure 5.6(a). The task consists of two sessions: during the practice session (0–1 s), the vehicle has access to the locations of all waypoints in the roadmap, providing critical guidance for path planning. This roadmap information, combined with sensory feedback, including angular and linear velocities and deviation from the centerline, constitutes the state received by the controller. In the race session (1–25 s), the roadmap information is no longer available, requiring the controller to rely on the stored roadmap information from the practice session to navigate. The goal of this task is to maximize the driving distance within a constrained timeframe of 25 seconds. Therefore, a structured and fine-grained memory of the roadmap is essential to avoid collisions and optimize driving strategies to achieve a high reward.

The model performance is evaluated through simulation on two representative tracks within the OpenAI Gym environment [224], utilizing realistic physics engines to simulate the uncertainties in real-world dynamic environments. We benchmark the performance of TE-SFNNs against vanilla feedforward SNN (SFNN) and LSTM using the on-policy reinforcement learning framework proximal policy optimization (PPO). The recorded rewards depicted in Figure 5.6(b) show that TE-SFNN-N consistently outperforms other models, achieving the highest rewards on both tracks. This superiority stems from TE-SNNs’ ability to encode critical temporal structures of the roadmap, enabling anticipation of upcoming changes and optimization of driving strategies. Taking advantage of this, the control commands and driving trajectories generated by TE-SFNNs are significantly smoother than those produced by SFNN and LSTM, as illustrated in Figures 5.6(c, d). These smoother commands are particularly advan-

tageous for real-world hardware implementations, as low-level actuator controllers often struggle to track abrupt variations in motor commands. By minimizing discrepancies between the ideal and executed commands, these smoother commands enhance stability and ensure more precise and reliable control.

Beyond superior driving performance, TE-SNNs also offer significant advantages in energy efficiency and robustness, which are critical for practical implementation in real-world autonomous systems operating under energy constraints and unpredictable perturbations. To demonstrate the energy efficiency of the proposed model, we develop an field-programmable gate array (FPGA)-based neuromorphic hardware system for TE-SNNs and compare their energy efficiency against that of LSTM models running on several conventional hardware platforms, including a GPU (NVIDIA RTX 3090), a CPU (Intel Xeon), and an edge AI accelerator designed for autonomous systems (Jetson Orin NX). Detailed hardware implementations are provided in Section 5.3.6. Energy efficiency is measured as throughput (inference steps per second) divided by average power consumption (watts). Results illustrated in Figure 5.6(e) show that TE-SFNNs achieve up to three orders of magnitude higher real-time inference energy efficiency compared to the best-performing LSTM baseline, highlighting their suitability for low-power, real-time autonomous driving systems. Furthermore, to evaluate the models' robustness in a real-world closed-loop scenario, we apply various noise perturbations from the previous section to the autonomous driving task (see Section 5.3.6 for details). As demonstrated in Figure 5.6(f), TE-SFNN maintains stable driving performance under various noise conditions, consistently achieving rewards above 60. In contrast, LSTM and SFNN exhibit increased instability in the presence of noise, resulting in lower rewards due to collisions. These results underscore the superior noise robustness of TE-SNNs in closed-loop control systems, highlighting their potential for deployment in real-world autonomous systems.

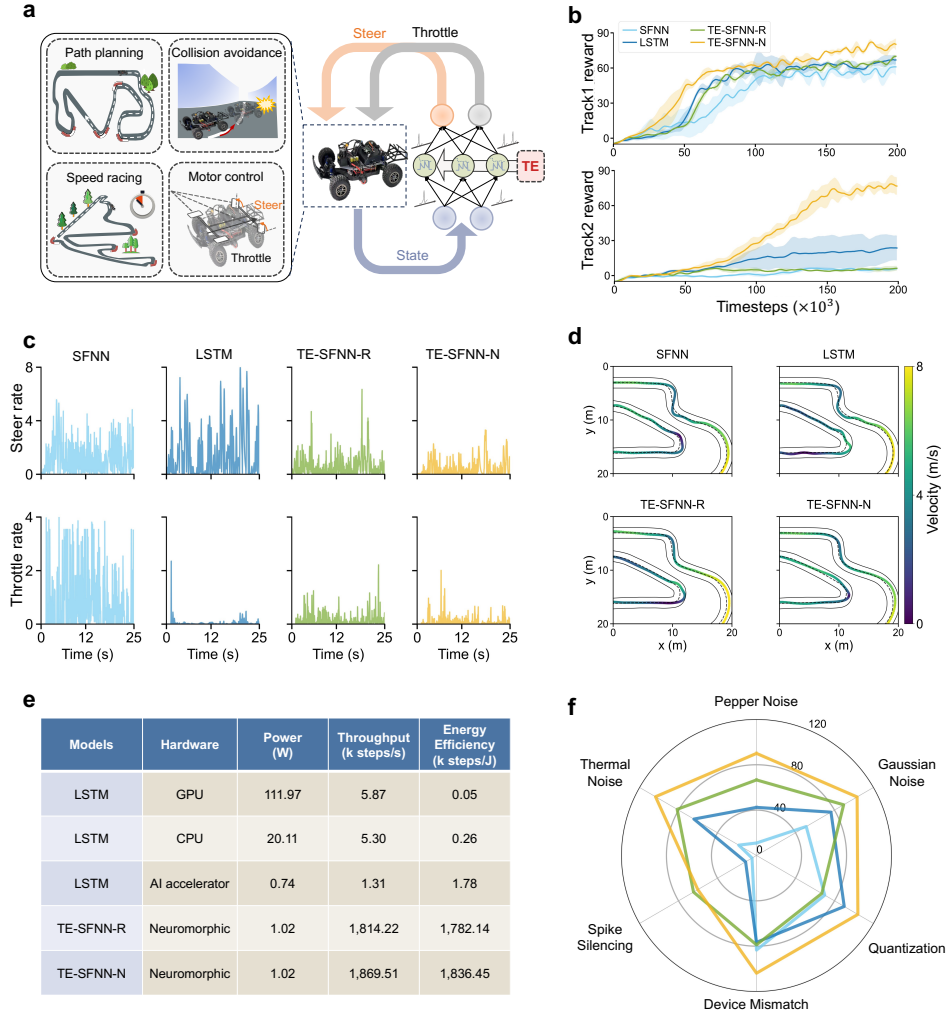


**Figure 5.6: Application of TE-SNNs in autonomous driving.** (a) Design of the autonomous driving task, which contains four subtasks: (1) Path planning – leveraging the available roadmap during the practice session to encode essential information for subsequent navigation; (2) Speed racing – maximizing travel distance within a constrained timeframe; (3) Collision avoidance – recalling roadmap information to optimize driving strategies and avoid obstacles; (4) Motor control – processing sensory feedback to generate real-time steering and throttle commands for stable control. (b) Reward evolution on two racing tracks, where the reward is proportional to the furthest distance achieved. (c) Comparison of motor command smoothness generated across models on Track 1. High steer and throttle rates indicate large variations in commands, which challenge low-level actuators and degrade performance. (d) Visualizations of the actual trajectories taken by different models on Track 1, where smoother paths and higher velocity reflect superior performance. (e) Comparison of power consumption, throughput, and energy efficiency between LSTM and TE-SFNNs deployed on different hardware platforms. (f) Average reward obtained by different models under six types of noise perturbations across five runs on Track 1, with radial distances representing rewards.

**Table 5.2: Impact of TE modulation designs in sequential image classification tasks.**

| ID | Design                                                                                                    | S-MNIST       | PS-MNIST      |
|----|-----------------------------------------------------------------------------------------------------------|---------------|---------------|
| 0  | Vanilla LIF                                                                                               | 51.11%        | 31.79%        |
| 1  | $V[t] = \alpha \cdot V[t - 1] + TE[t]$                                                                    | 83.79%        | 68.47%        |
| 2  | $V[t] = \alpha \cdot V[t - 1] + S[t - 1] \cdot TE[t]$                                                     | 95.08%        | 93.64%        |
| 3  | $V[t] = \alpha \cdot V[t - 1] + V[t - 1] \cdot TE[t]$                                                     | 95.11%        | 94.00%        |
| 4  | $\theta[t] = \beta \cdot (\theta[t - 1] - \theta_0) - TE[t] + \theta_0,$                                  | 91.64%        | 80.99%        |
| 5  | $\theta[t] = \beta \cdot (\theta[t - 1] - \theta_0) - TE[t] \cdot S[t - 1] + \theta_0,$                   | 96.13%        | 94.69%        |
| 6  | $\theta[t] = \beta \cdot (\theta[t - 1] - \theta_0) - \mathbf{TE}[t] \cdot \mathbf{V}[t - 1] + \theta_0,$ | <b>98.45%</b> | <b>98.24%</b> |

### 5.3.5 Impact of TE Modulation Designs

In the proposed TE-SNNs, the TE module is meticulously designed to integrate into neuron dynamics by modulating the firing threshold with the product of  $TE[t]$  and membrane potential  $V[t - 1]$ . To shed light on how this design affects the performance of TE-SNNs, we conduct an empirical ablation study to evaluate its effectiveness. To assess the impact of this design, we conduct an empirical ablation study using a three-layer fully connected TE-SFNN-N with 88 neurons per layer. The study systematically examines how TE interacts with neuronal states, either through output spikes  $S[t - 1]$  or membrane potential  $V[t - 1]$ , and whether its effect is incorporated into the membrane potential  $V[t]$  or the firing threshold  $\theta[t]$  of the LIF neuron to enhance excitability. As shown in Table 5.2, the proposed design consistently outperforms alternative configurations on two benchmarks, S-MNIST and PS-MNIST. A notable performance boost is observed when replacing  $S[t - 1]$  with  $V[t - 1]$ , highlighting the importance of leveraging membrane potential for temporal modulation. Further improvements are achieved by incorporating the multiplicative interaction between  $V[t - 1]$  and the temporal structure encoding  $TE[t]$  over time, effectively leveraging the inner product operation to quantify the alignment between  $V[t]$  tra-

jectories and the temporal structures encoded in  $TE[t]$ . Hence, this study provides compelling evidence regarding the effectiveness of the proposed modulation strategy.

### 5.3.6 Experimental Setups

This subsection describes the experimental setups, including task formulations, model configurations, and the hardware deployment used for energy evaluation.

#### Task Formulations

To evaluate the capacity of SNNs to capture fundamental temporal structures, including interval, duration, and periodicity, we meticulously designed three timing tasks: (1) In the Interval Discrimination task, each input sequence of length  $T = 100$  consists of eight consecutive spike events, where the interval lengths between adjacent spikes are randomly selected with equal probability from a longer interval  $\Delta t_1$  and a shorter interval  $\Delta t_2$ . The longer interval  $\Delta t_1$  is randomly sampled from 7 to 9 timesteps, while the shorter interval  $\Delta t_2$  is randomly sampled from 3 to 5 timesteps. The model’s objective is to discriminate distinct interval lengths within the spike train and, at the final timestep, output the cumulative count of the longer interval, denoted as  $\text{count}(\Delta t_1)$ . (2) In the Duration Discrimination Task, two spiking bursts with varied durations  $\Delta t_1$ ,  $\Delta t_2$  occur at random positions within an input sequence of length  $T = 100$ . The duration of each burst is randomly sampled from 5 to 20 timesteps. During each burst firing period, the sequence value is set to 1, while all other timesteps remain 0. The network is tasked with identifying the durations of the two spike bursts and computing their duration difference at the final timestep. (3) In the Synchronization task, the onset of the periodic stimulus is randomly distributed between timesteps 0 and 25, with a total sequence length of  $T = 100$ . Each periodic stimulus consists of four spikes occurring at a fixed ISI, which is uniformly sampled from 3 to 8. The network’s objective is to generate an output sequence that starts

in sync with the stimulus onset and continues until the end of the sequence while maintaining the same ISI.

The Delayed Recall task, designed to evaluate the capacity for processing long-term spatiotemporal structures, is an event-based variation of the classic copy memory problem [225]. The sequence consists of three stages: a 10-timestep initial stage, a  $(T-20)$ -timestep delay stage, and a 10-timestep recall stage, where the total sequence length is customizable as  $T$ . For each timestep in the initial stage, one of the input neurons indexed from 1 to 8 is randomly activated to generate a spike, each neuron can be activated repeatedly. The spike activities of these neurons form the input sequences. In the recall stage, the 9<sup>th</sup> input neuron exhibits burst firing behavior as a recall cue, prompting the output neurons to activate in the same order as the input sequence, thereby reconstructing the original sequence during this stage.

In the noise robustness evaluation demonstrated in Figure 5.5(a), we categorize hardware-related noise into three primary sources: synaptic noise, neuronal noise, and background noise, each further divided into six specific types: quantization, device mismatch, thermal, spike silencing, pepper, and spike jittering noises. Each type is applied at five progressively increasing levels (Level 1–5) to systematically assess model robustness under varying noise conditions. The detailed descriptions of each noise type are as follows:

- Quantization noise arises from the finite memory allocated to synaptic weights when deployed on neuromorphic hardware. We simulate this by applying post-training weight quantization, reducing the bit resolution across five levels: 8-bit (Level 1), 6-bit (Level 2), 5-bit (Level 3), 4-bit (Level 4), and 3-bit (Level 5). Lower-bit quantization introduces increasingly severe discretization errors, affecting the precision of synaptic weights.
- Device mismatch noise stems from the inherent variability in the fabrication process, leading to deviations between actual and desired synaptic parameters. We

model this by applying Gaussian noise to the initial weight values,  $\theta_0$ , such that perturbed weights follow a normal distribution:  $\theta \sim \mathcal{N}(\theta_0, \sigma\theta_0)$ . Here,  $\sigma$  represents the standard deviation, which linearly increases from 0.1 (Level 1) to 0.5 (Level 5), representing the growing severity of parameter deviations.

- Thermal noise is typically modeled as Gaussian noise, perturbing the neuronal input current  $I_0$  to follow a normal distribution  $I \sim \mathcal{N}(I_0, \sigma I_0)$ . In our experiment,  $\sigma$  varies from 0.1 (Level 1) to 0.5 (Level 5), representing an increasing level of thermal disturbances.
- Spike silencing noise occurs when a subset of spiking neurons fails to generate output spikes, degrading information transmission fidelity. To simulate this, we randomly mask a portion of neuron spike outputs, from 3% (Level 1) to 15% (Level 5), with higher levels leading to more extensive disruptions.
- Pepper noise appears as random white and black pixel distortions in frame-based image inputs. To simulate the presence of pepper noise, we introduce it into the image samples in S-MNIST, where the probability of noise occurrence is defined as  $p = \sigma \bar{I}$ . Here,  $\bar{I}$  represents the average intensity of the input image, and  $\sigma$  is the noise level, increasing from 0.1 (Level 1) to 0.5 (Level 5).
- Spike jittering noise distorts event timing in event-based datasets, a common transmission issue in neuromorphic chips [226]. To simulate this, we randomly shift a proportion of input spikes by 100 timesteps, with the percentage of affected spikes linearly increasing from 10% (Level 1) to 50% (Level 5).

The first five types of noise are applied to the S-MNIST dataset, while spike jittering noise is specifically tested on the SHD dataset, given its event-based nature. To ensure a comprehensive evaluation that reflects real-world noise conditions, models with identical architectures are trained on noise-free training samples and evaluated on noise-corrupted test samples.

The autonomous driving task, designed to evaluate models’ practicability to memorize and utilize temporal structures in real-world scenarios, is developed using the F1TENTH Gym environment [227]. This environment enables the simulation of a four-wheel vehicle with a realistic physics engine on predefined tracks, commonly used in reinforcement learning research. The Barcelona and YasMarina tracks are adopted for our evaluation [228]. At each timestep, the model receives a 106-dimensional state observation, which includes: yaw angle error, lateral distance error, steering angle, body linear velocity, body angular velocity, and a waypoint list indicating roadmap information. During the first second, the waypoint list consists of the relative positions of the next 50 waypoints, providing critical information for path planning. After this initial period, the roadmap information is removed by setting the waypoint values to zero, compelling the model to rely on its memory of future paths for effective navigation. The model outputs are decoded into steering and throttle commands, which directly control the vehicle actuators, enabling autonomous driving on the tracks.

For the noise robustness evaluation in the autonomous driving task, we introduce Gaussian noise  $s$  to the state observations, where  $s$  follows the normal distribution  $\mathcal{N}(0, \sigma)$ . Additionally, the first five noise types are implemented using the same methodologies as Figure 5.5(a). Specifically, quantization noise is applied using 8-bit precision, device mismatch noise is introduced with  $\sigma = 0.05$ , thermal noise with  $\sigma = 0.1$ , spike silencing is simulated by randomly masking 10% of the neuronal outputs, and pepper noise is applied with  $\sigma = 0.1$ .

## Model Configurations

For all experiments, the SNN architecture consists of stacked spiking hidden layers followed by a decoder layer. Following prior SNN studies [68, 132], the decoder layer is implemented as a non-spiking fully connected layer across all experiments. However, for visualization purposes in Figures 5.5(b, c), it is replaced with a spiking

layer to better illustrate network dynamics. The network output is derived from the decoder layer over time, with task-specific strategies determining how the output is processed. For the Interval Discrimination and Duration Discrimination tasks, the network output is taken from the decoder layer at the final timestep. For the S-MNIST, PS-MNIST, SHD, and GSC datasets, the network output is computed by averaging the decoder layer’s output over all timesteps. In contrast, for the Synchronization, Delayed Recall, PTB, and autonomous driving tasks, where labels vary over time, the output is dynamically determined at each timestep and passed through a task-specific activation function: Heaviside for Synchronization, Softmax for Delayed Recall and PTB, and Tanh for autonomous driving.

While the modulation strategy of the TE module has been primarily demonstrated using the LIF neuron model, it is inherently flexible and extensible to a broad range of spiking neuron models. To this end, we apply the TE modulation strategy to three representative spiking neuron models. Table 5.3 summarizes the original dynamics and their TE-modulated counterparts for each neuron model. In the table,  $V[t]$  denotes the soma membrane potential,  $I[t]$  is the input current,  $S[t]$  is the binary spike output determined by the Heaviside function  $H(\cdot)$ , and  $\alpha$  represents the decay factor that controls how fast  $V[t]$  leaks over time. In all cases, the core structure of the model is preserved, while the firing threshold  $\theta[t]$  is dynamically modulated by the TE signal, enabling the neuron to adapt its excitability in a temporally structured manner.

- **PLIF neuron** [68] extends the standard LIF model by introducing a learnable parameter  $a$ , which is transformed via a sigmoid function  $\sigma(\cdot)$  to produce a neuron-specific decay factor  $\alpha$ . This enables heterogeneous information retention across neurons, enhancing the model’s capacity to process multi-scale temporal features.
- **CLIF neuron** [75] enhances the standard LIF model by introducing an additional gating variable  $m[t]$  to regulate the integration and retention of membrane potential

**Table 5.3: Comparison of vanilla and TE-modulated neuronal dynamics across different neuron models.**

| Model  | Original Neuronal Dynamics                                                                                                                                                                                                                         | TE-Modulated Neuronal Dynamics                                                                                                                                                                                                                                                                                                             |
|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PLIF   | $\begin{cases} \alpha = \sigma(a), \\ V[t] = \alpha \cdot V[t-1] \cdot (1 - S[t-1]) + I[t], \\ S[t] = H(V[t] - \theta). \end{cases}$                                                                                                               | $\begin{cases} \theta[t] = \beta \cdot (\theta[t-1] - \theta_0) - TE[t] \cdot V[t-1] + \theta_0, \\ \alpha = \sigma(a), \\ V[t] = \alpha \cdot V[t-1] \cdot (1 - S[t-1]) + I[t], \\ S[t] = H(V[t] - \theta[t]). \end{cases}$                                                                                                               |
| CLIF   | $\begin{cases} V[t] = \alpha \cdot V[t-1] + I[t], \\ S[t] = H(V[t] - \theta), \\ m[t] = m[t-1] \cdot \sigma((1 - \alpha) \cdot V[t] + S[t]), \\ V[t] = V[t] - S[t] \cdot (\theta + \sigma(m[t])). \end{cases}$                                     | $\begin{cases} \theta[t] = \beta \cdot (\theta[t-1] - \theta_0) - TE[t] \cdot V[t-1] + \theta_0, \\ V[t] = \alpha \cdot V[t-1] + I[t], \\ S[t] = H(V[t] - \theta[t]), \\ m[t] = m[t-1] \cdot \sigma((1 - \alpha) \cdot V[t] + S[t]), \\ V[t] = V[t] - S[t] \cdot (\theta[t] + \sigma(m[t])). \end{cases}$                                  |
| DH-LIF | $\begin{cases} \alpha = \sigma(a), \quad \gamma = \sigma(b), \\ D[t] = \gamma \cdot D[t-1] + (1 - \gamma) \cdot I[t], \\ V[t] = \alpha \cdot V[t-1] \cdot (1 - S[t-1]) + (1 - \alpha) \cdot R \cdot D[t], \\ S[t] = H(V[t] - \theta), \end{cases}$ | $\begin{cases} \theta[t] = \beta \cdot (\theta[t-1] - \theta_0) - TE[t] \cdot V[t-1] + \theta_0, \\ \alpha = \sigma(a), \quad \gamma = \sigma(b), \\ D[t] = \gamma \cdot D[t-1] + (1 - \gamma) \cdot I[t], \\ V[t] = \alpha \cdot V[t-1] \cdot (1 - S[t-1]) + (1 - \alpha) \cdot R \cdot D[t], \\ S[t] = H(V[t] - \theta[t]), \end{cases}$ |

$V[t]$ , which improves the neuron’s capacity to model long-term dependencies and extract temporal features.

- **DH-LIF neuron** [47] extends the standard LIF model by incorporating heterogeneous dendritic substructures, where each soma is connected to multiple dendritic branches. Each branch maintains its own membrane potential  $D[t]$ , while the soma integrates these branch signals using readout weights  $R$ . The somatic decay factor  $\alpha$  and dendritic decay factor  $\gamma$  are controlled by learnable parameters  $a$  and  $b$ , respectively, which are optimized during training to enable more flexible and diverse temporal integration. In our implementation, we adopt two dendritic branches.

## Hardware Deployment and Energy Efficiency Evaluation Setups

To validate the energy efficiency of TE-SNNs in real-world deployment scenarios, we compare their performance on a closed-loop autonomous driving task against a

standard LSTM model. The TE-SNNs are deployed on an FPGA-based neuromorphic chip, while the LSTM is evaluated on three representative AI computing platforms: a CPU, a GPU, and an edge AI accelerator. In the following, we detail the hardware specifications, deployment configurations, and measurement protocols.

The hardware specifications and software environments used for deployment on each platform are listed below:

- **GPU:** NVIDIA RTX 3090 GPU with 24 GB, running on a server equipped with Ubuntu 20.04.6 and Intel(R) Xeon(R) Platinum 8350C CPU. The model is implemented using Pytorch 2.0.1 and CUDA 11.8.
- **CPU:** Intel(R) Xeon(R) Platinum 8350C CPU (32 cores, 64 threads). Inference is executed using Pytorch 2.0.1 in CPU mode. CUDA is disabled during CPU measurements.
- **AI accelerator:** NVIDIA Jetson Orin NX (8-core ARM Cortex + 1024-core NVIDIA Ampere architecture GPU with 32 Tensor Core + 16GB 128-bit LPDDR5). All models are deployed using PyTorch 2.1.0 with the JetPack 5.1.5 software environment. The device supports four predefined power modes (10W, 15W, 25W, and MAXN). We evaluate all configurations and identify the highest energy efficiency under the 15W setting.
- **Neuromorphic chip:** AX7350B FPGA development board (ZYNQ XC7Z035FFG676 SoC with 171,900 LUTs, 343,800 FFs, 500 BRAMs, and 900 DSPs). The dedicated hardware processor is designed at the RTL level, with Verilog code synthesized and implemented using the Xilinx Vivado 2021.1 Design Suite.

We evaluate all models on the autonomous driving task, where each model receives a 106-dimensional input vector representing state observations at each timestep and produces a 32-dimensional driving command vector as output. To ensure a fair comparison, all models are configured with a single hidden layer comprising 64 units.

- **LSTM-based controller:** Input (106)  $\rightarrow$  LSTM (106  $\rightarrow$  64)  $\rightarrow$  Linear (64  $\rightarrow$  32)  $\rightarrow$  Output (32),
- **TE-SNN-based controller:** Input (106)  $\rightarrow$  Linear (106  $\rightarrow$  64)  $\rightarrow$  TE-LIF (64)  $\rightarrow$  Linear (64  $\rightarrow$  32)  $\rightarrow$  Output (32).

The baseline LSTM model is deployed on three platforms using PyTorch with float32 precision. In contrast, we developed an FPGA-based neuromorphic hardware for TE-SFNN-R and TE-SFNN-N. The deployment adopts the Address-Event Representation (AER) format for inter-layer communication, where each AER packet consists of 24 bits. As illustrated in Figure 5.7, each layer is composed of four core modules:

- **Decoder:** It unpacks incoming AER packets to extract the current timestamp, source neuron index, and input data;
- **Controller:** It manages the overall system state and orchestrates data flow across modules;
- **TE neuron array:** It includes multiple TE neurons, each equipped with an arithmetic unit, a neurodynamics unit, and local buffers for weights and TE dynamics;
- **Encoder:** It collects the spikes generated by the TE neurons and converts them back into AER packets for transmission to the next layer.

When a layer receives an AER packet, the decoder first unpacks the 24-bit data into a 9-bit timestamp, a 7-bit source neuron index, and an 8-bit input data. Within the TE neuron array, each TE neuron utilizes the timestamp to retrieve the corresponding 24-bit modulation value  $TE[t]$  from a dedicated TE dynamics buffer, which is implemented as a lookup table indexed by timestep. Simultaneously, the source neuron index is used to retrieve the associated 8-bit synaptic weight from a weight buffer, also realized as a lookup table with neuron index as address and synaptic weight as

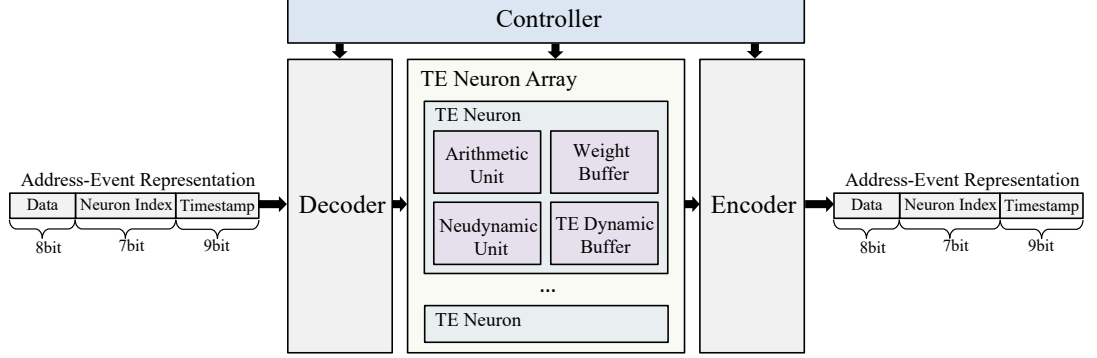


**Figure 5.7: FPGA-based neuromorphic hardware designed for TE-SNNs.** The TE-SNN models are deployed on FPGA-based neuromorphic hardware implemented on the ZYNQ XC7Z035FFG676 platform, operating at 200 MHz. Each spiking layer comprises four core modules: a decoder (parsing incoming 24-bit AER packets into timestamp, neuron index, and input data), a controller (managing system state and timing), a TE neuron array (responsible for neuronal state update and spike generation), and an encoder (converting output spikes into AER packets for the next layer). Hardware-level throughput and power consumption are estimated using Vivado implementation tools, reporting a throughput of 1814.22k steps/s for TE-SNN-R and 1869.51k steps/s for TE-SNN-N. Both TE-SNN models consume 1.02 W of power, with resource utilization of 6.60% LUTs, 2.96% FFs, 15.00% BRAMs, and 7.11% DSPs.

stored value. These inputs are then forwarded to the arithmetic unit, where membrane potential accumulation is performed. Subsequently, the neurodynamics unit carries out threshold adaptation according to the TE modulation signals. Following this, spike generation and membrane potential decay are performed. Any resulting spikes are encoded by the encoder into new AER packets, which are transmitted to the subsequent layer. Throughout the entire process, all data routing and timing operations are governed by the controller, which centrally manages state transitions and enables efficient pipelined execution of the entire layer. The number of TE neurons per layer and the overall network depth are fully configurable via parameter settings, allowing flexible architectural customization based on task requirements.

The energy efficiency used in the evaluation is defined as:

$$\text{Energy Efficiency (k steps / J)} = \frac{\text{Throughput (k steps / s)}}{\text{Power Consumption (W)}}, \quad (5.15)$$

where throughput denotes the number of decision steps processed per second, and power consumption refers to the average system power (watts) during the inference process.

To fairly assess model-specific power consumption and throughput across different hardware platforms, we first execute a closed-loop real-time inference for each model to generate a specific observation sequence reflecting its control behavior in the autonomous driving task. Each observation sequence is then replayed as input to its corresponding model for open-loop inference during energy efficiency measurements, eliminating model-irrelevant energy contributions arising from environment interaction and simulation overhead. All measurements also exclude overhead from model loading and data preprocessing, ensuring accurate evaluation of execution performance.

For all devices on which the LSTM baseline is deployed (i.e., GPU, CPU, and edge AI accelerator), we conduct a warm-up phase with 5 inference trials, followed by another 5 measured trials to compute average throughput and power consumption. This process is repeated three times, and the final results are obtained by averaging across runs. Specifically, throughput is obtained by dividing the total number of inference timesteps by the corresponding wall-clock duration. The power consumption is sampled at every simulation timestep using platform-specific tools:

- **GPU:** Power consumption of the GPU is monitored using NVIDIA’s ‘`nvidia-smi`’ utility, which provides real-time power draw in watts.
- **CPU:** For CPU power measurements, the Intel Running Average Power Limit (RAPL) interface is utilized to access energy consumption data.
- **AI accelerator:** Power is measured using the output of the onboard ‘`tegrastats`’ monitoring utility, which reports real-time voltage and current readings. Display output and peripheral devices are disabled during measurement to minimize extraneous energy consumption.

In contrast, TE-SNN models run on a dedicated FPGA-based neuromorphic chip. To verify the correctness of the hardware design, we first develop a functional simulator. Subsequently, we use Vivado simulation tools to measure the end-to-end inference latency and estimate power consumption from post-implementation reports. Throughput is computed as the reciprocal of inference latency.

## 5.4 Discussion and Conclusion

This chapter incorporates neural modulation mechanisms into spiking neuron models by introducing TE-SNNs, which integrate a brain-inspired TE module to modulate neuronal dynamics, thereby enhancing their temporal processing capability and robustness. TE-SNNs not only excel in timing tasks well-established in cognitive neuroscience, including discriminating intervals, durations, and periodicity, but also facilitate the establishment of long-range temporal dependencies crucial for real-world tasks. We demonstrate the exceptional temporal processing capabilities of TE-SNNs across various tasks, including visual classification, speech recognition, and language modeling, where they significantly outperform conventional SNNs and non-spiking LSTM models.

TE-SNN represents a fundamental shift from previous studies on temporal processing. Existing studies primarily focus on enhancing the temporal processing capabilities of spiking neurons through the introduction of slow-decaying state variables [35, 46, 72], heterogeneous state variables [21, 47, 71, 229], and the incorporation of axonal delays [212]. However, these methods predominantly aim to expand the state space of intrinsic neuronal variables, such as membrane potential and firing threshold. Due to the inherent decay and reset mechanisms governing these variables, critical temporal information is gradually lost over time, limiting their effectiveness at modeling long-term or highly dynamic temporal structures. In contrast, TE-SNNs leverage a permanent encoding space to encode essential temporal structures, effectively overcoming

this limitation. Consequently, TE-SNNs are particularly well-suited for real-world pattern recognition applications requiring complex, long-term context modeling.

TE-SNNs also provide valuable insights into the emergence of temporal structure encoding observed in biological neural systems. Specifically, we demonstrate that the TE module autonomously develops task-specific temporal structures. These structures underpin the formation of long-term memory, enabling the accurate recall of spatiotemporal patterns even after extended delays. This capability closely mirrors findings in neuroscience, where temporal structure encoding within regions like the hippocampus is fundamental to constructing structured memory representations that support sequential recall and temporal localization of past events [202, 217]. Furthermore, neurons within the trained TE-SNNs exhibit pronounced selectivity for specific temporal structures, manifested as burst-firing behavior triggered by target temporal patterns. This phenomenon directly parallels biological observations in both the auditory system [230] and the hippocampus [231], where neurons employ burst firing as a selective response mechanism to temporally structured inputs. Collectively, these features suggest TE-SNNs may capture core computational principles underlying temporal processing and memory formation in biological systems.

Biological neural systems operate under pervasive noise, arising from synaptic variability, metabolic fluctuations, and environmental stimuli, yet maintain robust functionality through evolved mechanisms. Mirroring this challenge, the deployment of SNNs on analog neuromorphic substrates is critically hindered by inherent device-level non-idealities [232], including synaptic weight variations, neuronal state instability, and background noise. Conventional approaches to mitigate these hardware imperfections typically rely on complex solutions: advanced fabrication processes to improve device uniformity, meticulous calibration routines to compensate for variations, and circuit-level redundancy to average out errors [9, 223]. These strategies, however, inevitably increase system complexity, chip area, and power consumption, counteracting key neuromorphic efficiency goals. In stark contrast, TE-SNNs demonstrate

intrinsic robustness to such perturbations. Their emergent burst-firing dynamics enable reliable information processing under noisy conditions without requiring additional hardware mitigation strategies or incurring associated overheads. This inherent robustness positions TE-SNNs as a computationally efficient and hardware-friendly computational model for mixed-signal neuromorphic systems.

Autonomous systems require efficient and robust decision-making in uncertain, dynamic environments, often constrained by sensor and environmental noise. Our results demonstrate that TE-SNNs effectively address these challenges. In the closed-loop autonomous driving task, TE-SNNs outperformed both LSTM and SNN baselines by leveraging structured temporal memory, and crucially, maintained stable performance even under substantial noise perturbations. Moreover, we successfully deployed TE-SNNs on an FPGA-based neuromorphic hardware, achieving fast inference with more than  $1000\times$  greater energy efficiency compared to edge-accelerated LSTM models. Given these advantages, TE-SNNs represent a highly promising neuromorphic solution for meeting the growing demand for efficient and reliable autonomous systems [233].

## Chapter 6

# Neuromorphic Sequential Arena: A Benchmark for Neuromorphic Temporal Processing

This chapter advances the evaluation of spiking neuron models with NSA, a comprehensive benchmarking suite that enables systematic evaluation and facilitates progress in spiking neuron model research. The NSA includes seven real-world temporal processing tasks from a diverse range of application scenarios, each capturing rich temporal dynamics across multiple timescales. Utilizing NSA, we systematically evaluate advanced spiking neuron models and neural architectures, delivering a comprehensive benchmark of task performance, training efficiency, memory footprint, and energy efficiency. Benchmarking results highlight the urgent need for model designs that can consistently deliver high performance across tasks with varying temporal complexities while maintaining low computational costs. NSA thus provides an effective, versatile, and application-oriented framework for consistent model evaluation, paving the way toward the development of accurate and robust neuromorphic pattern recognition systems.

## 6.1 Introduction

Temporal processing is a critical capacity for pattern recognition, enabling intelligent systems to interpret time-varying sensory signals and make accurate, timely decisions in dynamic environments. Neuromorphic computing holds immense potential for processing these signals with ultra-low energy consumption and low latency. SNNs, inspired by the computational principles of biological brains [17], serve as a cornerstone of neuromorphic computing. While SNNs have achieved performance on par with traditional ANNs in many static image classification tasks, demonstrating significantly improved energy efficiency and reduced latency [14, 15, 16, 234], their capacities in processing temporal signals remain inferior to those of ANNs. Recently, numerous SNN approaches have been proposed, making substantial progress in bridging this gap. These advancements include enriching neural dynamic heterogeneity [46, 47], increasing neuronal structural complexity [21, 235, 236], and developing temporal parallelization methods for improved training efficiency [79, 229].

Despite this progress, advancements in this research area are hindered by the absence of effective benchmarks. Existing neuromorphic benchmarks predominantly focus on visual classification [117, 122, 125] and keyword spotting [129, 131] tasks. While these benchmarks have been valuable over the past decade in advancing neuromorphic computing research, they play a limited role in fostering developments in neuromorphic temporal processing due to three primary limitations. First, performance on existing SNN benchmarks has largely saturated, and these benchmarks fail to capture the rich temporal dynamics inherent in real-world signals, which typically encompass multiple timescales. Second, existing benchmarks do not adequately represent the diverse range of temporal processing scenarios that closely align with the interests and objectives of neuromorphic computing research. Third, inconsistent comparisons across studies and the absence of evaluations on critical efficiency metrics result in biased assessments of model practicality.

To address these limitations, this chapter proposes a comprehensive benchmark called **NSA**, designed to establish an effective, versatile, and application-oriented evaluation framework for neuromorphic temporal processing. Firstly, NSA encompasses a broad range of tasks that reflect real-world temporal processing scenarios across application areas of significant interest to the neuromorphic research community, including human-computer interaction, speech processing, robotics, and biomedical applications. Furthermore, to ensure that these tasks possess an adequate level of temporal complexity suitable for SNNs, we introduce a novel tool for analyzing the temporal dependencies required to address a given task, referred to as the Segregated Temporal Probe (STP). This tool has been employed to validate the effectiveness of both commonly used neuromorphic datasets and the proposed NSA in benchmarking the temporal processing capacity of SNNs. Finally, using NSA, we conduct a comprehensive comparative study of recently introduced spiking neuron models and neural architectures in terms of task performance, training speed, memory usage, and energy efficiency. By providing a side-by-side comparison of these baselines, NSA serves as a valuable foundation for understanding the current status and practicality of advanced spiking neuron models. Collectively, NSA is designed as an evolving framework capable of integrating emerging neuromorphic algorithms, thereby fostering advancements toward more effective and efficient neuromorphic temporal processing systems.

The remainder of this chapter is organized as follows. In Section 6.2, we present the overall framework of NSA, outlining its design principles, task descriptions, and the STP tool. In Section 6.3, we first apply STP to evaluate the effectiveness of commonly used neuromorphic benchmarks and our proposed NSA. Next, we conduct a comprehensive benchmarking of representative spiking neuron models and neural architectures using NSA, comparing their performance, training speed, memory usage, and energy consumption. Finally, Section 6.4 summarizes the findings and implications of this chapter.

## 6.2 Neuromorphic Sequential Arena

The NSA comprises seven tasks that span a diverse array of real-world scenarios relevant to neuromorphic research. These tasks, characterized by varying levels of temporal complexity, serve as an effective benchmark for assessing the temporal processing capacities of different SNN approaches. In the following, we detail our design principles, task formulation, and the STP tool used to assess task effectiveness.

### 6.2.1 Design Principles

This subsection outlines the five foundational principles for the NSA design:

- **Neuromorphic relevance.** Tasks should highlight the advantages of neuromorphic solutions, such as energy efficiency, low latency, and robustness.
- **Temporal complexity.** Tasks should reflect rich temporal dynamics inherent in real-world scenarios, requiring models to establish temporal dependencies across multiple timescales. To measure this, we introduce a quantitative temporal analysis tool in the Section 6.2.3.
- **Challenging.** Tasks should exhibit an adequate level of difficulty, highlighting distinguishable performance among existing SNN approaches while offering significant opportunities for improvement. In quantitative terms, advanced neuromorphic models are expected to yield accuracy improvements exceeding 2% on these tasks.
- **Training resource.** Tasks should account for training time and GPU resource constraints to ensure accessibility for both researchers and practitioners. In particular, for SNN approaches that rely on temporally serial simulation, it is crucial that the training can be completed within 24 hours on a single GPU for 100 epochs.
- **Application versatility.** Tasks should encompass a wide range of real-world

application scenarios with distinct requirements and data characteristics, bridging the gap between theoretical advancements and practical applications.

### 6.2.2 Tasks

In the following, we will introduce the suite of tasks included in the proposed NSA benchmark, meticulously designed following the principles outlined above. We provide a comprehensive overview of these tasks, including detailed task descriptions, application scenarios, and specific capacities they evaluate. The task characteristics, evaluation metrics, and dataset configurations are summarized in Table 6.1. To facilitate benchmarking efforts, we provide a comprehensive **open-source library** publicly available at <https://github.com/liyc5929/neuroseqbench>. The library includes detailed data preparation guidelines, an end-to-end preprocessing pipeline, and one-command reproducibility of all experiments in this chapter. It also allows seamless integration of novel spiking neuron models, ensuring consistent and standardized evaluation across tasks.

#### Autonomous Localization (AL)

Due to the inherent characteristics of low latency and high energy efficiency, neuromorphic systems present considerable potential for robotic control. A fundamental challenge in this domain is predicting the current state of the robot after executing a sequence of actions, which is crucial for overcoming sensory feedback delays and promptly adapting to external perturbations. Addressing this challenge requires a neuromorphic system to establish long-term temporal dependencies, effectively modeling the intricate relationship between past actions and the present state.

Given the above requirement and the significance of this task, we propose a new synthetic dataset called AL. AL simulates a scenario in which a mobile robot must determine whether it is positioned on the left or right plane following a sequence

**Table 6.1: Summary of task characteristics, evaluation metrics, and other configurations in the proposed NSA benchmark.**

| Task   | Dataset | Sequence length | Metric | Backbone model     | Training samples | Testing samples |
|--------|---------|-----------------|--------|--------------------|------------------|-----------------|
| AL     | AL      | User-defined    | Acc.   | MLP                | 50,000           | 5,000           |
| HAR    | WISDM   | 200             | Acc.   | MLP                | 26,048           | 6,512           |
| EEG-MI | OpenBMI | 500             | Acc.   | MLP                | 17,280           | 4,320           |
| SSL    | SLoClas | 500             | Acc.   | MLP                | 37,426           | 8,969           |
| ALR    | DVS-Lip | 200             | Acc.   | MLP                | 14,896           | 4,975           |
| AD     | N-DNS   | 751/3,751       | SI-SNR | Spiking-FullSubNet | 60,000           | 341             |
| ASR    | AISHELL | 76–505          | CER    | VGG-MLP            | 360,294          | 7,176           |

of ordered actions, including ‘turn left’, ‘turn right’, ‘go straight’, and ‘stop’. This constitutes a binary classification task with customized sequence lengths and action distributions, facilitating an effective evaluation of the capacity of SNNs to establish temporal dependencies across various timescales.

### Human Activities Recognition (HAR)

The HAR task focuses on identifying human activity patterns from time-series data collected by wearable sensors. This task represents a significant application of neuromorphic computing, encompassing real-time robotic locomotion control, pose estimation, and surveillance, where accurate and efficient temporal processing is essential. HAR requires the modeling of fine-grained temporal trajectories of human activities embedded within noisy sensor signals. Therefore, HAR is particularly well-suited for evaluating the capacity of SNNs to capture complex short-term temporal dependencies and to demonstrate their robustness against unpredictable noise conditions.

In this task, we utilize data samples from the WISDM dataset [237], which includes gyroscope data collected from a smartwatch. The data is recorded at 20 Hz and segmented into 10-second intervals, resulting in a sequence length of 200. This task is intentionally challenging, requiring models to accurately classify 18 actions that may

exhibit subtle differences, such as ‘eating chips’ or ‘eating pizza’.

### **Electroencephalogram Motor Imagery (EEG-MI)**

We further evaluate the effectiveness of SNNs in brain-computer interfaces and real-time cognitive monitoring through the EEG-MI task. This task focuses on decoding motion imagery from EEG sequences, which is particularly challenging due to the sparse, noisy, and highly dynamic nature of EEG signals. Therefore, it serves as an effective benchmark for assessing the capacity of SNNs to capture both temporal and spatial dependencies with a high degree of robustness.

Specifically, we utilize motion imagery data from the OpenBMI dataset [238], which comprises 62-channel EEG recordings collected from 52 subjects at a sampling rate of 1 kHz. The model is tasked with performing a binary classification aimed at distinguishing between imagery trials of left- and right-hand grasping. To ensure manageable training costs, we downsample the sequence length to 500. To mitigate the risk of overfitting due to subject-specific bias, we employ a cross-trial validation approach, in which all samples are randomly partitioned into training and testing sets.

### **Sound Source Localization (SSL)**

Identifying the origin of a sound source within a noisy environment is a crucial survival skill for animals and holds significant importance for applications such as hearing aids and robotics. In this task, we aim to evaluate the temporal processing capacity and noise robustness of SNN approaches. To this end, we adopt the SLoClas dataset [153], which contains 4-channel audio recordings from a single sound source positioned at azimuth angles ranging from  $0^\circ$  to  $360^\circ$  in  $5^\circ$  increments, resulting in a 72-class classification problem. Additionally, directional background noise fragments are introduced to the raw audio signals at a challenging SNR ratio of 0. The resulting

noisy signals are segmented into sequences of length 500, then fed directly into SNNs without any spectral preprocessing. This task poses a specific challenge for temporal processing, as the model must learn to map the temporal delays between different audio channels to their corresponding azimuth angles in the presence of background noise.

### **Automatic Lip-Reading (ALR)**

ALR aims to recognize spoken words from a speaker’s lip movements and plays a pivotal role in various real-world applications, such as video surveillance and speech recognition in noisy environments. DVS cameras, characterized by high dynamic range and low latency, have emerged as ideal visual front-ends for capturing the fine-grained lip movements essential for the ALR task. Their event-based outputs are particularly well-suited for neuromorphic systems.

In this task, we adopt the DVS-Lip dataset [239], which consists of 100 spoken word classes captured by the DAVIS346 event camera from 40 individuals. The task is particularly challenging, as the training and testing sets include different individuals, requiring the SNN model to generalize across unseen speakers. Moreover, the output is decoded solely from the spiking activities generated at the final timestep, requiring models to extract and retain crucial spatiotemporal features over an extended period. To ensure manageable training costs, each sample is segmented into 200 temporal bins and center-cropped to a resolution of  $88 \times 88$  pixels.

### **Audio Denoising (AD)**

Removing noise from received audio signals to enhance overall quality is critical for many edge applications, such as hands-free communication and hearing aids. Given the low-power and real-time processing requirements of these tasks, neuromorphic solutions present a highly promising approach. Notably, among all tasks within the

proposed NSA benchmark, AD stands out as a regression problem, providing a unique opportunity to evaluate the expressive power of SNN models in capturing subtle and continuous temporal variations.

In this task, we utilize the publicly available Intel Neuromorphic Deep Noise Suppression (N-DNS) Challenge dataset [240]. This dataset comprises clean speech samples in multiple languages (i.e., English, German, French, Spanish, and Russian) as well as noise samples collected from diverse acoustic environments. We employ the official synthesizer script to generate a 495-hour training subset and a 5-hour validation subset. All audio samples are synthesized at a 16 kHz sampling rate and segmented into a uniform length of 30 seconds. We perform audio preprocessing steps in accordance with the winning entry of the N-DNS Challenge [241] and adopt their model architecture as the default. For model evaluation, we adopt the scale-invariant signal-to-noise ratio (SI-SNR) metric.

### **Automatic Speech Recognition (ASR)**

ASR transcribes spoken language from audio into text, serving as a foundational technology for various applications, including voice assistants, transcription services, and speech translation tools. This task presents significant challenges for SNNs due to varying sequence lengths, speaker characteristics, and acoustic conditions.

In this task, we utilize the AISHELL dataset [242], an open-sourced Mandarin speech corpus comprising approximately 170 hours of speech data from 400 speakers, encompassing a wide range of accents and speaking styles. Its moderate data scale and high speaker diversity render it an effective testbed for assessing the model’s capacity to handle linguistic and acoustic variations. Model performance is evaluated using the character error rate (CER).

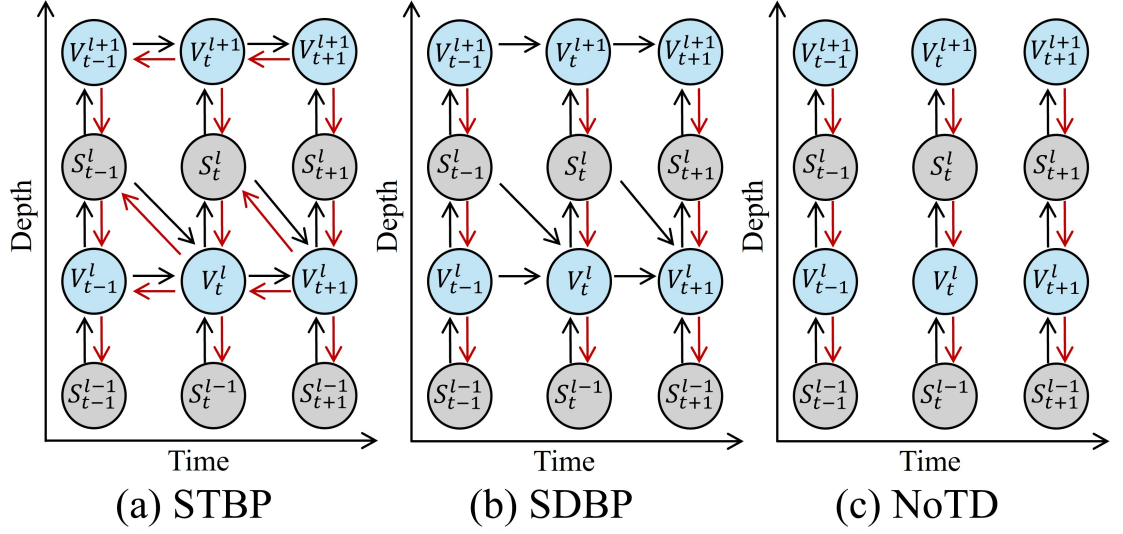


Figure 6.1: Comparison of the three training algorithms in STP.

### 6.2.3 Segregated Temporal Probe

To elucidate the limited effectiveness of commonly used neuromorphic benchmarks in evaluating the temporal processing capacity of SNNs and to demonstrate the efficacy of NSA, we introduce STP. STP assesses the contributions of establishing temporal dependencies by quantifying the impact of isolating forward and backward temporal processing pathways of spiking neurons on task performance. As illustrated in Figure 6.1, STP consists of three evaluation modules, each corresponding to a specific training algorithm: Spatio-Temporal Backpropagation (STBP) [24], Spatial Domain Backpropagation (SDBP), and No Temporal Domain (NoTD). Details of these three algorithms are outlined below, using the LIF neuron model [51] as an example.

The LIF neuron accumulates input spikes  $\mathbf{S}^{l-1}[t]$  from the preceding layer  $l - 1$  into its membrane potential  $\mathbf{V}^l[t]$ . When  $\mathbf{V}^l[t]$  surpasses a threshold  $\theta$ , it triggers a spike and subsequently resets to the resting potential. The dynamics of a LIF neuron can be formulated as:

$$\mathbf{V}^l[t] = \underbrace{\alpha \mathbf{V}^l[t-1](1 - \mathbf{S}^l[t-1])}_{\text{Forward temporal propagation}} + \mathcal{W}^l \mathbf{S}^{l-1}[t], \quad (6.1)$$

$$\mathbf{S}^l[t] = H(\mathbf{V}^l[t] - \theta), \quad (6.2)$$

where  $\alpha$  determines the decay rate of  $\mathbf{V}^l[t]$  over time,  $\mathcal{W}^l$  is the synaptic weight matrix, and  $H(\cdot)$  is the Heaviside step function. The recursive update in Eq. (6.1) allows the LIF neuron to iteratively propagate temporal information from  $\mathbf{V}^l[t-1]$  to  $\mathbf{V}^l[t]$ , enabling the integration of temporal information over time.

**STBP** preserves this temporal propagation in both forward and backward passes, where the gradient of the loss with respect to weights is computed as:

$$\frac{\partial \mathcal{L}}{\partial \mathcal{W}^l} = \sum_{t=1}^T \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]} \frac{\partial \mathbf{V}^l[t]}{\partial \mathcal{W}^l} = \sum_{t=1}^T \boldsymbol{\delta}^l[t] \mathbf{S}^{l-1}[t]^\top, \quad (6.3)$$

$$\boldsymbol{\delta}^l[t] = \underbrace{\boldsymbol{\delta}^l[t+1] \frac{\partial \mathbf{V}^l[t+1]}{\partial \mathbf{V}^l[t]}}_{\text{Backward temporal propagation}} + \boldsymbol{\delta}^{l+1}[t] \frac{\partial \mathbf{V}^{l+1}[t]}{\partial \mathbf{V}^l[t]} \quad (6.4)$$

where  $\boldsymbol{\delta}^l[t] \triangleq \frac{\partial \mathcal{L}}{\partial \mathbf{V}^l[t]}$ , which is the backpropagated gradient error along both spatial and temporal dimensions.

**SDBP** only retains the temporal propagation in the forward pass but omits its impact on the backward pass. This prevents gradients from propagating across time, thereby limiting the learning of temporal dependency. The modified gradient error is:

$$\hat{\boldsymbol{\delta}}^l[t] = \begin{cases} \frac{\partial \mathcal{L}}{\partial \mathbf{V}^L[t]}, & l = L, \\ \hat{\boldsymbol{\delta}}^{l+1}[t] \frac{\partial \mathbf{V}^{l+1}[t]}{\partial \mathbf{V}^l[t]}, & l < L. \end{cases} \quad (6.5)$$

**NoTD** removes temporal propagation in both forward and backward passes by processing each timestep independently, simplifying neuronal dynamics as:

$$\mathbf{V}^l[t] = \mathcal{W}^l \mathbf{S}^{l-1}[t], \quad (6.6)$$

which excludes the update of  $\mathbf{V}^l[t]$  compared to Eq. (6.1), rendering it unable to integrate any temporal information across time. The gradient in NoTD is the same as Eq. (6.5) in SDBP.

By training an SNN using the three algorithms described and comparing their performance gaps, we can quantify the effectiveness of a specific task in assessing the temporal processing capacity of the SNN:

1. A similar performance between STBP and SDBP indicates that the task fails to leverage temporal propagation in the backward pass to establish meaningful temporal dependencies, and, therefore, is ineffective in evaluating temporal processing capacity.
2. A comparable performance between NoTD and STBP implies that the task can be solved without leveraging any temporal information, making it unsuitable as a benchmark for temporal processing.
3. In contrast, significant performance degradation in SDBP and NoTD compared to STBP indicates that the task contains rich temporal information and can effectively evaluate temporal processing capacity.

## 6.3 Experimental Results

In this section, we begin by outlining the detailed preprocessing techniques tailored for each NSA task to ensure consistent comparisons. Subsequently, we employ the STP tool to validate the effectiveness of both existing neuromorphic benchmarks and the proposed NSA in evaluating the temporal processing capacity. Next, we conduct a comprehensive benchmarking of recently proposed spiking neuron models and neural architectures using NSA, thereby illuminating the current landscape of SNNs in temporal processing. Additionally, we incorporate deployment-critical effi-

ciency metrics to further benchmark these models, including training speed, memory usage, and energy consumption, to assess their practicality for real-world applications. For all tables in this chapter, **bold values** indicate the best performance, and underlined values represent the second best.

### 6.3.1 Data Preprocessing

- **AL:** In this task, we utilize our proposed synthetic dataset AL. Each sample begins with a robot positioned at coordinate  $(0, 0)$ , facing the positive y-axis. The robot then executes a randomly generated sequence of actions, with each action sampled from the set {'turn left', 'turn right', 'go straight', 'stop'} according to the probabilities  $[0.05, 0.05, 0.45, 0.45]$ , respectively. At the end of the sequence, the robot's final x-coordinate determines the classification:  $x \leq 0$  is "left", otherwise "right".
- **HAR:** The smartwatch gyroscope data (sampled at 20 Hz) is segmented using a sliding window approach. Each window has a length of 200 timesteps, with a hop length of 100 timesteps. The data features angular velocity in three coordinates (X, Y, Z). To eliminate subject-specific bias and ensure consistency, data from all subjects is z-score normalized per coordinate, shuffled, and split into training and testing sets.
- **EEG-MI:** EEG motor imagery data is preprocessed using a widely adopted EEG signal preprocessing pipeline [243]. The raw signals are bandpass-filtered (0.5–80 Hz), downsampled to 100 Hz, and cleaned using Independent Component Analysis (ICA) to remove artifacts. The data are then segmented to extract the principal EEG data relevant to the task. Subsequently, each channel is z-score normalized to ensure feature consistency. Finally, the samples are shuffled across subjects and split into training and testing sets to eliminate subject-specific biases and ensure effective model evaluation.

- **SSL:** We preprocess the raw audio signals (sampled at 16 kHz) using a sliding window approach with a window size of 500 and a hop size of 500. To simulate real-world application scenarios, directional background noise from  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ , and  $270^\circ$  provided in the dataset are randomly segmented and added to each sample at 0 SNR. Notably, raw time-domain signals are processed directly, without the application of any frequency spectral analysis.
- **ALR:** We preprocess the raw event data by converting asynchronous events into frame-based representations. Specifically, events are grouped into their nearest temporal bins, forming  $T$  temporal bins, where  $T = 200$  in our experiments. To preserve polarity information, positive and negative events are mapped into two separate channels. For data augmentation, we apply a series of transformations to the training data, including center cropping and random cropping to achieve an  $88 \times 88$  spatial resolution, horizontal flipping with a probability of 0.5, cutout, and random zoom-in/zoom-out. For the test data, only center cropping is applied.
- **AD:** All audio samples are processed at a sampling rate of 16 kHz and standardized to a fixed duration of 30 seconds. For audio clips shorter than 30 seconds, we concatenate additional speech segments from the same speaker, inserting a 0.2-second silence between utterances to maintain natural pauses. Noisy audio is generated by mixing clean speech with randomly selected noise samples at signal-to-noise ratios ranging from -5 dB to 20 dB, ensuring a diverse range of noise conditions. To simulate varying input loudness levels, we apply loudness normalization to each noisy sample, adjusting levels between -35 dB and -15 dB relative to full scale. For spectral analysis, the short-time Fourier transform (STFT) is configured with a 32 ms window length (512 samples) and an 8 ms hop length (128 samples), using a Hanning window and a 512-point Fast Fourier Transform to capture detailed frequency information.

- **ASR:** Experiments are conducted using the ESPnet [244] toolkit, following its standard recipes for data preparation, model training, and evaluation. This ensures a consistent and reproducible experimental setup.

### 6.3.2 Benchmark Effectiveness Validation

We first apply the proposed STP tool to evaluate the effectiveness of 12 widely adopted pattern recognition benchmarks in the neuromorphic community. These benchmarks can be broadly categorized into three types: static image recognition, event-based vision, and keyword spotting.

For static image datasets including MNIST [117], CIFAR-10 [118], and CIFAR-100 [118], inputs are constructed by repeating the same image across timesteps, thus lacking intrinsic temporal dynamics. This limitation is evidenced by our experimental results. As shown in Table 6.2, NoTD and SDBP achieve accuracy comparable to STBP, demonstrating that modeling temporal relationships is unnecessary for high performance on these datasets.

Event-based vision datasets such as including N-MNIST [121], CIFAR10-DVS [122], DVS-Gesture [125], DVS-SLR [127], GAIT-DAY-DVS [126], and HARDVS [128] contain sparse event-driven temporal dynamics. However, the results in Table 6.5 reveal that NoTD performs similarly to or even outperforms STBP, indicating that these datasets can be effectively addressed at the frame level without the establishment of temporal dependency.

For keyword spotting tasks, including GSC [129], SHD [131], and SSC [131], the results in Table 6.4 reveal only a marginal performance gap between STDP and STBP, along with moderate accuracy achieved by NoTD. These findings suggest that temporal propagation primarily contributes to the integration of short-term features during the forward pass, while backward temporal propagation has a negligible impact on establishing temporal dependencies. Consequently, these benchmarks are inadequate

**Table 6.2: Validating the effectiveness of static image recognition datasets in evaluating the temporal processing capacity of SNNs.**

| Method | MNIST               | CIFAR-10            | CIFAR-100           |
|--------|---------------------|---------------------|---------------------|
|        | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ |
| STBP   | <b>99.40</b>        | <b>94.86</b>        | <b>74.57</b>        |
| SDBP   | 99.27 (-0.13)       | 94.74 (-0.12)       | 74.35 (-0.22)       |
| NoTD   | 99.18 (-0.22)       | 93.46 (-1.40)       | 73.28 (-1.29)       |

**Table 6.3: Validating the effectiveness of event-based vision datasets in evaluating the temporal processing capacity of SNNs.**

| Method | N-MNIST             | CIFAR10-DVS          | DVS-Gesture          | DVS-SLR             | GAIT-DAY-DVS         | HARDVS              |
|--------|---------------------|----------------------|----------------------|---------------------|----------------------|---------------------|
|        | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$  | Acc. (%) $\uparrow$  | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$  | Acc. (%) $\uparrow$ |
| STBP   | <b>99.49</b>        | 78.50                | 95.14                | <b>76.57</b>        | 93.52                | <b>48.64</b>        |
| SDBP   | 99.48 (-0.01)       | 79.00 (+0.50)        | <b>95.83</b> (+0.69) | 75.92 (-0.65)       | <b>93.70</b> (+0.18) | 48.01 (-0.63)       |
| NoTD   | 99.09 (-0.40)       | <b>80.00</b> (+1.50) | 94.44 (-0.70)        | 75.18 (-1.39)       | 93.47 (-0.05)        | 47.98 (-0.66)       |

**Table 6.4: Validating the effectiveness of keyword spotting datasets in evaluating the temporal processing capacity of SNNs.**

| Method | GSC                 | SHD                 | SSC                 |
|--------|---------------------|---------------------|---------------------|
|        | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ |
| STBP   | <b>92.91</b>        | <b>86.48</b>        | <b>67.13</b>        |
| SDBP   | 89.00 (-3.91)       | 85.07 (-1.41)       | 66.03 (-1.10)       |
| NoTD   | 77.53 (-15.38)      | 68.51 (-17.97)      | 44.97 (-22.16)      |

for effectively evaluating the temporal processing capacities of SNNs.

These findings underscore the limitations of existing benchmarks and the urgent need for a more effective neuromorphic temporal processing benchmark. NSA addresses these gaps by incorporating tasks that contain rich, multi-timescale temporal dependencies, ensuring an effective and consistent assessment of SNNs’ temporal processing capacities. To further validate its effectiveness, we apply the same STP-based analysis to the NSA benchmark. As shown in Table 6.5, both SDBP and NoTD exhibit

**Table 6.5: Validating the effectiveness of NSA in evaluating the temporal processing capacity of SNNs.**

| Method | AL                  | HAR                 | EEG-MI              | SSL                 | ALR                 | AD                     | ASR                  |
|--------|---------------------|---------------------|---------------------|---------------------|---------------------|------------------------|----------------------|
|        | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | Acc. (%) $\uparrow$ | SI-SNR (dB) $\uparrow$ | CER (%) $\downarrow$ |
| STBP   | <b>63.52</b>        | <b>81.27</b>        | <b>65.20</b>        | <b>8.88</b>         | <b>17.83</b>        | <b>11.47</b>           | <b>20.70</b>         |
| SDBP   | 58.52 (-5.00)       | 76.29 (-4.98)       | 57.48 (-7.72)       | 5.79 (-3.09)        | 2.47 (-15.36)       | 10.18 (-1.29)          | 26.30 (+5.60)        |
| NoTD   | 53.34 (-10.18)      | 67.75 (-13.52)      | 52.48 (-12.72)      | 3.45 (-5.43)        | 2.01 (-15.82)       | 9.47 (-2.00)           | 27.10 (+6.40)        |

substantial performance degradation compared to STBP across all tasks in NSA. This suggests that the seven selected tasks contain essential temporal dependencies that must be effectively captured to attain high performance. Consequently, the proposed NSA serves as a more effective benchmark for neuromorphic temporal processing.

### 6.3.3 Performance Benchmarking of SNN Models

To elucidate the current status of SNN models in temporal processing, we further evaluate five spiking neuron models notable for their enhanced temporal processing capacities: LIF, Context Embedding LIF (CE-LIF) [245], LTC [74], SPSN [79], and PMSN [229]. In addition to the commonly used SFNN and SRNN networks [35], we also benchmark five advanced neural architectures that excel in temporal processing, including Gated Spiking Neuron (GSN) [241], Spiking Temporal Convolutional Network (TCN) [246], Spike-Driven Transformer [181], Binary S4D [182], and Gated Spiking Unit (GSU) [182]. To ensure a fair comparison, all models are configured with a comparable number of trainable parameters for each task. For architectures like Spiking TCN and Spike-Driven Transformer, which are originally designed to repeat each timestep  $D$  times to enhance their model representation power, we set  $D = 1$  to facilitate fair comparisons with other models.

As shown in Table 6.6, our experimental results demonstrate that the performance of spiking neuron models varies significantly across tasks. LTC shows only marginal

**Table 6.6: Results of NSA benchmark for different spiking neuron models with SFNN or SRNN architectures.**

| Architecture | Neuron model | AL           | HAR          | EEG-MI       | SSL          | ALR          | AD            | ASR          | Average rank ↓ |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|---------------|--------------|----------------|
|              |              | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | SI-SNR (dB) ↑ | CER (%) ↓    |                |
| SFNN         | LIF          | 54.40        | 81.27        | 65.20        | 8.88         | 17.83        | 11.47         | 20.70        | 6.6            |
|              | CE-LIF       | 59.58        | 79.71        | 70.56        | 12.21        | 48.32        | -*            | 17.70        | 5.3            |
|              | LTC          | 77.40        | 80.97        | 64.42        | 10.33        | 48.93        | <b>14.36</b>  | 15.90        | 4.3            |
|              | SPSN         | 72.02        | <u>86.73</u> | <b>76.46</b> | <u>20.30</u> | 45.73        | 13.00         | 18.50        | 3.7            |
|              | PMSN         | <b>87.42</b> | <b>88.31</b> | <u>75.00</u> | <b>75.35</b> | <b>57.43</b> | <u>14.35</u>  | 17.70        | <b>1.9</b>     |
| SRNN         | LIF          | 56.50        | 77.32        | 64.68        | 6.17         | 34.71        | 9.36          | <u>15.70</u> | 6.4            |
|              | CE-LIF       | 60.62        | 80.83        | 74.35        | 12.72        | 51.64        | -*            | <b>15.30</b> | 3.7            |
|              | LTC          | <u>79.04</u> | 82.03        | 69.16        | 16.37        | <u>56.64</u> | 14.29         | 16.30        | <u>3.1</u>     |

\* These models are not applicable due to the inconsistent sequence lengths between the training and test phases.

improvements in noise-intensive tasks such as HAR, EEG-MI, and SSL, attributable to its input-dependent mechanism, which is highly susceptible to noise accumulation over time. In contrast, PMSN and SPSN neurons exhibit stronger noise robustness in these tasks. Additionally, PMSN and LTC excel in AL and ALR tasks requiring long-term memory, showcasing superior temporal retention capacities over extended periods. Furthermore, LTC and PMSN demonstrate high expressiveness in the AD task, while CE-LIF with recurrent connections stands out in the ASR task, exhibiting remarkable generalizability. Overall, PMSN emerges as the top-performing spiking neuron model on NSA, striking a balance between robustness, temporal dependency establishment, and generalizability across tasks.

For neural architectures, the results demonstrated in Table 6.7 show that Binary S4D and its variant GSU exhibit strong noise robustness in HAR, EEG-MI, and SSL tasks, outperforming other architectures in handling noisy inputs. In contrast, the GSN model achieves notable gains in AD and ASR tasks, which require high model expressiveness and generalizability. This result suggests the high effectiveness of GSN in capturing diverse, fine-grained temporal patterns. Conversely, SRNNs and Spike-Driven Transformer fall short in performance compared to other models. We noticed that SRNNs often suffer from training instability and convergence issues in our

**Table 6.7: Results of NSA benchmark for different neural architectures using LIF neurons.**

| Architecture             | AL           | HAR          | EEG-MI       | SSL          | ALR          | AD            | ASR          | Average rank ↓ |
|--------------------------|--------------|--------------|--------------|--------------|--------------|---------------|--------------|----------------|
|                          | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | Acc. (%) ↑   | SI-SNR (dB) ↑ | CER (%) ↓    |                |
| SFNN                     | 54.40        | 81.27        | 65.20        | 8.88         | 17.83        | 11.47         | 20.70        | 5.9            |
| SRNN                     | 56.50        | 77.32        | 64.68        | 6.17         | 34.71        | 9.36          | 15.70        | 5.9            |
| GSN                      | 67.22        | 82.31        | 68.85        | 10.34        | 21.17        | <b>14.45</b>  | <b>14.30</b> | 3.6            |
| Spiking TCN              | 69.88        | 82.41        | 75.58        | 40.60        | <b>47.14</b> | 12.77         | 17.10        | 3.1            |
| Spike-Driven Transformer | 58.00        | 71.07        | 69.12        | 5.75         | 39.62        | 9.86          | 36.80        | 5.7            |
| Binary S4D               | <b>81.44</b> | <b>89.37</b> | <b>78.61</b> | <u>79.34</u> | <u>44.80</u> | <u>14.21</u>  | 15.20        | <b>1.7</b>     |
| GSU                      | <u>80.42</u> | <u>89.16</u> | <u>77.43</u> | <b>82.39</b> | 41.35        | 14.05         | <u>14.40</u> | <u>2.1</u>     |

experiments, resulting in comparable or worse performance than SFNNs. The Spike-Driven Transformer struggles to establish temporal dependencies effectively due to its binary activations (i.e.,  $D = 1$ ), which significantly constrains its expressiveness. Meanwhile, this architecture is typically designed to work with large datasets and networks, the moderate dataset size of NSA and the small parameter count in our evaluation may constrain the full exploitation of its architectural advantages.

### 6.3.4 Efficiency Benchmarking of SNN Models

We further report the efficiency metrics of the evaluated SNN models on the AL task, including training speed, GPU memory usage, and energy efficiency, which are critical factors for practical implementation. To ensure a fair comparison, all evaluations are conducted using a batch size of 256 and all evaluated SNN models are configured with a uniform network dimension, comprising two hidden layers with 256 channels each. Training speed and memory costs are evaluated with sequence lengths of {200, 400, 800}, while energy efficiency is assessed with a sequence length of 400. The comparative results are summarized in Table 6.8.

**Table 6.8: Comparison of SNN models in terms of the training speed, memory consumption, and energy efficiency.** Values in brackets indicate ratios relative to the baseline LIF-SFNN model.

| Architecture             | Neuron model | Training speed (k step/s) $\uparrow$ |                   |                   | Memory consumption (GB) $\downarrow$ |                   |                   | Energy efficiency    |                       |                         |
|--------------------------|--------------|--------------------------------------|-------------------|-------------------|--------------------------------------|-------------------|-------------------|----------------------|-----------------------|-------------------------|
|                          |              | 200                                  | 400               | 800               | 200                                  | 400               | 800               | ACs (k) $\downarrow$ | MACs (k) $\downarrow$ | Power (mJ) $\downarrow$ |
| SFNN                     | LIF          | 1.91 (1.0)                           | 1.91 (1.0)        | 2.04 (1.0)        | <b>0.49 (1.0)</b>                    | <b>0.98 (1.0)</b> | <b>1.96 (1.0)</b> | 3.72 (1.0)           | 0.26 (1.0)            | <b>4.52 (1.0)</b>       |
|                          | CE-LIF       | 1.25 (0.7)                           | 1.31 (0.7)        | 1.37 (0.7)        | 0.59 (1.2)                           | 1.18 (1.2)        | 2.35 (1.2)        | 3.22 (0.9)           | 0.77 (3.0)            | <u>6.43 (1.4)</u>       |
|                          | LTC          | 0.63 (0.3)                           | 0.66 (0.3)        | 0.68 (0.3)        | 1.43 (2.9)                           | 2.84 (2.9)        | 5.68 (2.9)        | <b>0.93 (0.3)</b>    | 262.91 (1,011)        | 1,210.23 (268)          |
|                          | SPSN         | 5.04 (2.6)                           | 4.14 (2.2)        | 3.21 (1.6)        | <u>0.50 (1.0)</u>                    | <u>1.01 (1.0)</u> | <u>2.01 (1.0)</u> | <u>1.56 (0.4)</u>    | 32.77 (126)           | 152.14 (33)             |
|                          | PMSN         | <u>6.34 (3.3)</u>                    | <u>6.31 (3.3)</u> | <u>6.24 (3.1)</u> | 1.23 (2.5)                           | 2.46 (2.5)        | 4.92 (2.5)        | 10.27 (2.8)          | 4.66 (18)             | 30.68 (6.8)             |
| SRNN                     | LIF          | 1.19 (0.6)                           | 1.25 (0.7)        | 1.30 (0.6)        | 0.54 (1.1)                           | 1.08 (1.1)        | 2.16 (1.1)        | 22.91 (6.2)          | 0.26 (1.0)            | 21.79 (4.8)             |
|                          | CE-LIF       | 0.92 (0.5)                           | 0.98 (0.5)        | 1.00 (0.5)        | 0.64 (1.3)                           | 1.28 (1.3)        | 2.55 (1.3)        | 8.26 (2.2)           | 0.77 (3.0)            | 10.96 (2.4)             |
|                          | LTC          | 0.55 (0.3)                           | 0.57 (0.3)        | 0.57 (0.3)        | 1.43 (2.9)                           | 2.84 (2.9)        | 5.68 (2.9)        | 2.12 (0.6)           | 262.91 (1,011)        | 1,211.30 (268)          |
| GSN                      | LIF          | 0.88 (0.5)                           | 0.92 (0.5)        | 0.92 (0.5)        | 0.64 (1.3)                           | 1.28 (1.3)        | 2.55 (1.3)        | 27.33 (7.3)          | 1.28 (4.9)            | 30.48 (6.7)             |
| Spiking TCN              |              | 4.92 (2.6)                           | 4.99 (2.6)        | 5.01 (2.5)        | 0.66 (1.4)                           | 1.30 (1.3)        | 2.58 (1.3)        | 50.49 (11)           | <b>0.00 (0.0)</b>     | 56.10 (15)              |
| Spike-Driven Transformer |              | 1.86 (1.0)                           | 1.99 (1.0)        | 2.06 (1.0)        | 3.07 (6.3)                           | 6.10 (6.2)        | 12.16 (6.2)       | 214.70 (58)          | <b>0.00 (0.0)</b>     | 193.22 (42)             |
| Binary S4D               |              | <b>7.04 (3.7)</b>                    | <b>7.07 (3.7)</b> | <b>7.33 (3.6)</b> | 1.26 (2.6)                           | 2.51 (2.6)        | 5.00 (2.6)        | 44.49 (12)           | 5.43 (21)             | 65.01 (14)              |
| GSU                      |              | 5.93 (3.1)                           | 5.91 (3.1)        | 5.92 (2.9)        | 1.48 (3.0)                           | 2.95 (3.0)        | 5.88 (3.0)        | 6.37 (1.7)           | 4.92 (19)             | 28.36 (6.3)             |

## Training Speed

Our evaluation results on NSA exhibit notable training speed differences between serial and parallel models. Serial models, including LIF, CE-LIF, LTC, and GSN, process temporal data sequentially, maintaining consistent but inherently slow training speeds per timestep, regardless of sequence lengths. Consequently, their total training time scales linearly with the sequence length, leading to high computational costs for processing long sequences. This limitation becomes even more pronounced when involving complex neuronal dynamics. For instance, LIF achieves the fastest training speed among serial models, whereas LTC achieves the slowest, highlighting that inefficient training in serial models poses a significant bottleneck for long sequence processing.

Parallel models, on the contrary, demonstrate high computational efficiency by processing data in multiple timesteps simultaneously. Our results show that recently

proposed parallel models, including SPSN, PMSN, Spiking TCN, binary S4D, and GSU, achieve approximately  $3\times$  speedup compared to serial models. However, the Spike-Driven Transformer fails to achieve noticeable speedups over the LIF model due to its high computational complexity. Additionally, the training speed of parallel models varies with sequence length, reflecting differences in their underlying parallelization strategies. Specifically, SPSN exhibits a decline in training speed as the sequence length increases, resulting in an exponentially growing total training time. This slowdown arises from the design of its receptive kernel, which expands proportionally with sequence length to capture long-range temporal dependencies. Consequently, the neuron dynamics incur a quadratic time complexity of  $\mathcal{O}(L^2)$ . In contrast, other parallel models sustain consistent training speeds across varying sequence lengths, owing to their more efficient linear-time complexity of  $\mathcal{O}(L)$ .

### Memory Usage

Despite enhanced temporal processing capacity offered by many recent SNN models, our finding reveals that these advancements come with a significant increase in memory consumption. Specifically, while parallel computing models accelerate training speeds, they typically incur substantially higher memory usage, trading off between space and time. This inefficiency arises from their need to store additional states to capture temporal dependencies in parallel. Among these parallel models, the Spike-Driven Transformer stands out as the most memory-intensive due to the expanded embedding space involved in the self-attention mechanism. In contrast, PMSN, S4D, and GSU consume comparatively less memory, benefiting from their compact representation of model states. However, they still exceed the memory requirements of serial approaches due to the additional storage needed for their parallelized operations. SPSN employs a 1-D temporal convolution kernel to capture local temporal features without the need to buffer extra transient states, making it a highly memory-efficient parallel architecture comparable to the LIF model. In contrast, serial models

generally consume less memory as they store fewer transient states across time. The only exception is LTC, whose poor memory efficiency stems from its complex computation graph involved in gating computations. Notably, memory consumption for all evaluated models scales linearly with sequence length, regardless of structure or computational strategy. It highlights SNNs’ memory efficiency in handling extended temporal sequences, as each timestep contributes a fixed and predictable amount of memory without exponential growth.

### Energy Efficiency

The high energy efficiency of SNNs promises to enable efficient temporal processing at the edge. To present a comprehensive assessment of the energy efficiency of advanced SNN models, we conduct a quantitative analysis of the number of MAC and AC operations per inference timestep and per sample. Additionally, we derive the average empirical energy consumption for each model based on the experimental data.

Specifically, we first derive theoretical energy consumption formulas based on the temporal dynamics and architectural structure of each model, as summarized in Table 6.9. In these formulations,  $E_{MAC}$  and  $E_{AC}$  represent the energy required for each MAC and AC operation on neuromorphic hardware, respectively.  $m$ ,  $n$ , and  $k$  denote the input size, hidden size, and kernel size, respectively.  $h$  is the hidden dimension of the feedforward module in Spike-Driven Transformer. The variable  $Fr_i$  captures the spike frequency of a spike sequence  $i$ , with examples including  $Fr_{in}$  and  $Fr_{out}$ , which represent the sparsity of input and output spike sequences for a given layer. For models with additional architectural components, such as the Spiking TCN,  $Fr_{conv2}$  specifies the sparsity of input spikes in the second convolution layer. Similarly, in the Spike-Driven Transformer,  $Fr_q$ ,  $Fr_k$ ,  $Fr_v$ ,  $Fr_{attn}$ ,  $Fr_{fc1}$ , and  $Fr_{fc2}$  denote the sparsity of LIF neuron populations in the self-attention and feedforward modules. For GSU,  $Fr_y$  and  $Fr_w$  represent the sparsity of ternary outputs and weights, respec-

**Table 6.9: Theoretical and empirical energy costs of different architectures and neurons.**

| Architecture             | Neuron model | Theoretical cost                                                                        | Empirical cost (nJ) |
|--------------------------|--------------|-----------------------------------------------------------------------------------------|---------------------|
| SFNN                     | LIF          | $(mnFr_{in} + nFr_{out})E_{AC}$<br>$+nE_{MAC}$                                          | 4.52                |
|                          | CE-LIF       | $(mnFr_{in} + nFr_{out})E_{AC}$<br>$+3nE_{MAC}$                                         | 6.43                |
|                          | LTC          | $(mnFr_{in} + 2nFr_{out} + 2n)E_{AC}$<br>$+(4nn + 3n)E_{MAC}$                           | 1,210.23            |
|                          | SPSN         | $mnFr_{in}E_{AC}$<br>$+nkE_{MAC}$                                                       | 152.14              |
|                          | PMSN         | $mnFr_{in}E_{AC}$<br>$+(2n \log_2(T) + 5n)E_{MAC}$                                      | 30.68               |
| SRNN                     | LIF          | $(mnFr_{in} + (nn + n)Fr_{out})E_{AC}$<br>$+nE_{MAC}$                                   | 21.79               |
|                          | CE-LIF       | $(mnFr_{in} + (nn + n)Fr_{out})E_{AC}$<br>$+3nE_{MAC}$                                  | 10.96               |
|                          | LTC          | $(mnFr_{in} + (nn + 2n)Fr_{out} + 2n)E_{AC}$<br>$+(4nn + 3n)E_{MAC}$                    | 1,211.3             |
| GSN                      | LIF          | $(2mnFr_{in} + 2nnFr_{out})E_{AC}$<br>$+5nE_{MAC}$                                      | 30.48               |
| Spiking TCN              |              | $(kmnFr_{in} + knnFr_{conv2})E_{AC}$                                                    | 56.10               |
| Spike-Driven Transformer |              | $((3Fr_{in} + Fr_{attn})nn + (Fr_qFr_k + Fr_v)nT$<br>$+ (Fr_{fc1} + Fr_{fc2})nh)E_{AC}$ | 193.22              |
| Binary S4D               |              | $2nnFr_{out}E_{AC}$<br>$+(2n \log_2(T) + 8n)E_{MAC}$                                    | 65.01               |
| GSU                      |              | $(nn(Fr_y + Fr_w) + 2n)E_{AC}$<br>$+(2n \log_2(T) + 6n)E_{MAC}$                         | 28.36               |

tively. To obtain accurate values for these spike frequencies, we perform inference on well-trained models using the AL task with a sequence length of 400. All models are configured with two hidden layers of 256 neurons each, and the second hidden layer is consistently used for cross-model comparisons. During inference, we measure the average spike frequency per timestep, per sample, and per neuron across all involved spiking populations. For empirical energy estimation, we adopt hardware-level energy coefficients derived from a 45 nm CMOS process [194], the energy per operation is set to  $E_{AC} = 0.9$  pJ for AC operations and  $E_{MAC} = 4.6$  pJ for MAC operations. These

values are then substituted into the theoretical formulas to calculate the empirical energy consumption for each model.

The results presented in Table 6.8 suggest that most advanced SNN models improve temporal processing capacity at the cost of increased energy consumption. Among the evaluated models, CE-LIF, PMSN, and GSU stand out for achieving a favorable trade-off between performance and energy efficiency, making them more suitable for deployment in resource-constrained edge systems. It is noteworthy that LTC presents two orders of magnitude higher energy consumption than standard LIF models, which can be attributed to the hundreds of thousands of MAC operations required for computing its temporal dynamics. Such high energy costs, comparable to traditional ANN methods, severely limit its practicality for energy-constrained systems. These observations highlight the critical role of efficiency evaluation in model design. To unleash the full potential of neuromorphic systems in temporal processing, we argue that the development of SNN models must prioritize not only the enhancement of task accuracy but also the maintenance of ultra-low energy consumption, thereby ensuring their competitiveness in real-world applications.

## 6.4 Discussion and Conclusion

This chapter advances the evaluation of spiking neuron models with NSA, an effective, versatile, and application-oriented benchmark designed to comprehensively assess the temporal processing capacities of SNNs across diverse application scenarios. To ensure rigorous and reliable assessment, we integrate a temporal dependency analysis tool, STP, into NSA to quantify the effectiveness of the benchmark. Our comparative analysis underscores both the progress and the challenges in neuromorphic temporal processing. While advanced spiking neuron models and neural architectures demonstrate remarkable improvements in task performance, many of them struggle to maintain efficiency in training speed, memory usage, and energy consumption,

which are crucial constraints for real-world applications. Our findings highlight the urgent need to develop effective SNN models capable of robustly processing temporal data while maintaining high energy efficiency. While this chapter provides a limited evaluation of SNN approaches due to time and resource constraints, we encourage the community to expand the scope of evaluations using NSA.

We envision NSA as an effective and adaptive benchmarking framework capable of addressing the evolving needs of the community. We hope this benchmark will inspire further advancements in neuromorphic temporal processing research, thereby facilitating the development of more accurate and robust neuromorphic pattern recognition systems. To support this vision and maximize NSA’s long-term impact, we are actively collaborating with NeuroBench [247], an open international benchmarking initiative that aims to enable community-wide, reproducible, and comparable evaluations under a unified framework. In addition, we plan to maintain the NSA repository by incorporating newly published models and tasks, as well as by encouraging open contributions from the research community.

## Chapter 7

# Conclusion and Future Directions

The human brain comprises approximately 86 billion neurons, operating in a highly parallel and event-driven manner to support a wide spectrum of neural functions. These neurons employ diverse coding schemes, exhibit complex internal structures, and undergo dynamic modulation, collectively enabling effective information processing across varying spatiotemporal scales. Inspired by these biological principles, this thesis focuses on enhancing SNNs with advanced spiking neuron models for accurate and robust pattern recognition. Our efforts center around four key dimensions: enriched information representation strategies, enhanced internal structural complexity, novel neural modulation mechanisms, and effective evaluation benchmarks. Specifically, we propose a hybrid neural coding and learning framework that improves the fidelity, efficiency, robustness, and adaptability of neural representations in spiking neurons; a multi-compartment neuron model that enhances expressiveness while supporting efficient parallel computation of spiking neurons; a temporal modulation mechanism that augments the temporal structure awareness of spiking neurons; and a neuromorphic benchmarking suite that enables systematic evaluation and advances research on spiking neuron models. Collectively, these innovations narrow the gap between artificial spiking neuron models and biological neurons, and lay a principled

foundation for developing accurate and robust neuromorphic pattern recognition systems. This chapter summarizes the core contributions of this thesis and outlines several avenues for extending this line of research toward greater adaptability and efficiency.

### 7.1 Summary of Contributions

In **Chapter 3**, this thesis proposes a hybrid neural coding and learning framework that advances spiking neuron models with enriched information representation strategies. This framework takes inspiration from the rich diversity of neural coding strategies observed in biological neural systems, each exhibiting distinct advantages under different task demands and environmental conditions. To emulate this, the proposed framework incorporates a neural coding zoo formed by a collection of brain-inspired neural coding schemes. According to the assignment strategy and task requirements, these coding schemes can be flexibly allocated to different network layers. Moreover, we introduce a layer-wise learning method to ensure the desired neural representation can be effectively achieved. Consequently, the hybrid coding SNNs could take advantage of diversified coding schemes to achieve optimal task performance. We validate the effectiveness of our framework on both image and sound localization tasks. The experimental results demonstrate that the proposed hybrid coding SNNs exhibit the desired attributes of high accuracy, robustness to noise, low inference latency, and superior energy efficiency. As an early attempt in this direction, the successful implementation of the proposed hybrid neural coding approaches underscores the potential of heterogeneous spiking neuron models for building adaptable neuromorphic systems.

In **Chapter 4**, this thesis proposes a PMSN model that advances spiking neuron models with intricate internal structures. Inspired by the compartmental organization of biological neurons, PMSN integrates multiple interactive compartments to

capture and integrate temporal features across a wide range of timescales, addressing one of the core limitations of simplified point models. To address the significant constraint on training speed caused by the increased complexity of neuronal dynamics, we further introduce two temporal parallelization techniques that decouple the temporal dependencies, enabling efficient parallel computation on modern GPU-accelerated platforms. Both theoretical analysis and visualizations of temporal dynamics are provided to substantiate the efficacy of the PMSN model in establishing temporal dependencies across various timescales. Moreover, the proposed PMSN model offers a remarkable training acceleration of over  $10\times$  compared to conventional LIF neurons. Extensive experiments across various pattern recognition benchmarks demonstrate that the proposed model consistently outperforms prior SOTA spiking neuron models in accuracy and training speed, while maintaining comparable computational cost. This advancement opens up new opportunities for solving challenging real-world pattern recognition tasks by unleashing the potential of detailed spiking neuron models.

In **Chapter 5**, this thesis proposes TE-SNNs that advance spiking neuron models with novel neural modulation mechanisms. Motivated by the temporal encoding in biological neurons, the TE module is designed in two complementary forms: rhythmic TE, which captures periodic temporal regularities, and non-rhythmic TE, which models task-specific temporal structures. The TE module temporally modulates the firing excitability of spiking neurons, resulting in burst spikes when inputs align with the encoded structures, thereby enabling effective and robust processing of complex temporal signals. TE-SNNs not only excel in timing tasks investigated in neurocognitive experiments, including discriminating intervals, durations, and periodicity, but also facilitate the establishment of long-range temporal dependencies. Extensive experiments across visual, speech, and textual benchmarks consistently demonstrate that TE-SNNs achieve superior accuracy and robustness compared to baseline models. In a real-world autonomous driving scenario, TE-SNNs further exhibit practical advantages by generating stable driving policies, while offering substantial gains in energy

efficiency and robustness to hardware non-idealities. Collectively, TE-SNNs provide a promising neuromorphic solution to address the increasing demand for accurate and robust real-time pattern recognition systems.

In **Chapter 6**, this thesis proposes the NSA, which advances the evaluation of spiking neuron models with a comprehensive, application-oriented framework. NSA encompasses seven representative tasks from diverse real-world neuromorphic application scenarios, including human-computer interaction, speech recognition, robotics, and biomedical signal processing. These tasks exhibit rich temporal dynamics and reflect the practical challenges that neuromorphic systems are expected to address in real-world deployments. To validate the effectiveness of NSA in evaluating the temporal modeling capacity of spiking models, we develop the STP, a novel tool that quantitatively assesses the degree of temporal dependency inherent in datasets. Finally, we conduct a comprehensive comparative study of recently introduced spiking neuron models and neural architectures on NSA in terms of task performance, training speed, memory usage, and energy efficiency. By providing a side-by-side comparison, NSA offers valuable insight into the performance and applicability of current spiking models in temporal processing. As an evolving evaluation framework, NSA is designed to accommodate future advances in spiking neuron models and foster the development of more accurate and robust neuromorphic pattern recognition systems.

## 7.2 Future Work

Taking inspiration from biological principles, this thesis has advanced spiking neuron models to achieve higher accuracy and robustness in real-world pattern recognition tasks. Nevertheless, significant opportunities remain for further improvements. Building on the contributions presented here, this thesis outlines several avenues for extending this line of research toward greater adaptability and efficiency, thereby moving artificial neural systems closer to the functional capabilities of biological in-

telligence:

In Chapter 3, we introduce a hybrid neural coding framework that integrates diverse coding schemes observed in biological systems. However, the current assignment of coding schemes across network layers is manually configured and lacks adaptability. In contrast, biological systems have evolved highly optimized combinations of coding schemes through millions of years of adaptation, achieving extraordinary flexibility across diverse tasks and environments. To emulate this remarkable adaptivity, future work may explore multi-objective evolutionary algorithms to automatically optimize neural coding configurations with respect to various objectives, including accuracy, robustness, energy consumption, and inference latency. Furthermore, inspired by Mixture-of-Experts (MoE) architectures that underpin many recent large-language-model (LLM) architectures, a promising direction is to introduce a learnable dynamic routing mechanism that automatically assigns neural coding schemes to different network layers or blocks. Such a router could jointly optimize coding allocation with network training, enabling stable, adaptive, and context-aware inference under varying environments and task requirements.

In Chapters 4 and 5, we incorporate brain-inspired multi-compartment structures and temporal modulation mechanisms into spiking neuron models. Although these advancements mark an essential step toward more accurate and robust spiking neuron models, the intricate structural and functional properties of biological neurons, such as diversity in dendritic morphology, cell types, and synaptic connectivity, remain largely unexplored and offer promising directions for future research. Future efforts should focus on constructing hierarchical spiking models that reflect biological systems' structural and functional heterogeneity, while also addressing the computational and deployment challenges associated with such complexity to ensure a balance between biological plausibility and practical feasibility.

In Chapter 4, we propose two parallelization strategies to accelerate SNN training by decoupling temporal dependencies among neuronal states, thereby enabling efficient

temporal parallelism on modern GPU frameworks. However, neuronal dynamics that are more biologically plausible often involve nonlinear state transitions, which inherently limit the extent to which temporal decoupling can be applied. Future work may explore advanced algorithms capable of approximating complex neuronal dynamics while enabling parallel computation, or develop custom hardware primitives and scheduling strategies that support efficient acceleration. These efforts could ultimately enable large-scale brain simulations and the construction of neuromorphic large models (LMs) for real-world pattern recognition tasks at scale.

On the evaluation front, Chapter 6 establishes a standardized and extensible framework for benchmarking spiking models on pattern recognition tasks. However, existing benchmarking toolchains are often developed in isolation, supporting only narrow subsets of models or tasks, and adopting incompatible interfaces and evaluation metrics. This fragmentation limits the ability to establish a unified neuromorphic benchmarking ecosystem, thereby hindering reproducibility and impeding systematic progress tracking. A critical future direction is to promote cross-community collaboration to align model interfaces, data pipelines, and evaluation protocols, enabling seamless benchmarking across platforms and fostering transparency and comparability.

One critical future direction lies in the system-level co-design of spiking neuron models. Most existing research has concentrated on isolated optimizations, targeting neural coding schemes, internal structures, neural modulations, or synaptic plasticity rules, while overlooking their potential synergies. In biological systems, however, these components have been shaped through evolution as interdependent mechanisms, jointly refined to support adaptive and efficient information processing. Inspired by this evolutionary co-optimization, future efforts should move beyond modular optimization and pursue holistic co-design approaches that jointly optimize these interdependent mechanisms within a unified framework.

Finally, from a hardware perspective, the growing complexity of spiking neuron mod-

els necessitates more flexible deployment solutions. However, most existing neuromorphic chips adopt fixed architectural configurations, such as memory hierarchies and interconnects, which are difficult to modify once fabricated. This rigidity hinders their compatibility with evolving neuron models and diverse task requirements. To address this limitation, future neuromorphic hardware should support sparse, event-driven inter-neuronal communication while enabling reconfigurable, fine-grained intra-neuronal computation, thereby flexibly adapting to evolving neuron model dynamics and task-specific demands.

# References

- [1] S. Theodoridis and K. Koutroumbas, *Pattern recognition*. Elsevier, 2006.
- [2] N. Maslej, L. Fattorini, R. Perrault, Y. Gil, V. Parli, N. Kariuki, E. Capstick, A. Reuel, E. Brynjolfsson, J. Etchemendy *et al.*, “Artificial intelligence index report 2025,” *arXiv preprint arXiv:2504.07139*, 2025.
- [3] W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski, *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.
- [4] P. Poirazi, T. Brannon, and B. W. Mel, “Pyramidal neuron as two-layer neural network,” *Neuron*, vol. 37, no. 6, pp. 989–999, 2003.
- [5] P. Poirazi and A. Papoutsis, “Illuminating dendritic function with computational models,” *Nature Reviews Neuroscience*, vol. 21, no. 6, pp. 303–321, 2020.
- [6] C. Koch, *Biophysics of computation: information processing in single neurons*. Oxford University Press, 2004.
- [7] J. J. DiCarlo, D. Zoccolan, and N. C. Rust, “How does the brain solve visual object recognition?” *Neuron*, vol. 73, no. 3, pp. 415–434, 2012.
- [8] N. Mesgarani and E. F. Chang, “Selective cortical representation of attended speaker in multi-talker speech perception,” *Nature*, vol. 485, no. 7397, pp. 233–236, 2012.

- 
- [9] D. Marković, A. Mizrahi, D. Querlioz, and J. Grollier, “Physics for neuromorphic computing,” *Nature Reviews Physics*, vol. 2, no. 9, pp. 499–510, 2020.
- [10] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, “Opportunities for neuromorphic computing algorithms and applications,” *Nature Computational Science*, vol. 2, no. 1, pp. 10–19, 2022.
- [11] D. Kudithipudi, C. Schuman, C. M. Vineyard, T. Pandit, C. Merkel, R. Kubendran, J. B. Aimone, G. Orchard, C. Mayr, R. Benosman *et al.*, “Neuromorphic computing at scale,” *Nature*, vol. 637, no. 8047, pp. 801–812, 2025.
- [12] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, “The spinnaker project,” *Proceedings of the IEEE*, vol. 102, no. 5, pp. 652–665, 2014.
- [13] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam *et al.*, “Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip,” *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.
- [14] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain *et al.*, “Loihi: A neuromorphic manycore processor with on-chip learning,” *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.
- [15] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He *et al.*, “Towards artificial general intelligence with hybrid tianjic chip architecture,” *Nature*, vol. 572, no. 7767, pp. 106–111, 2019.
- [16] D. Ma, X. Jin, S. Sun, Y. Li, X. Wu, Y. Hu, F. Yang, H. Tang, X. Zhu, P. Lin *et al.*, “Darwin3: a large-scale neuromorphic chip with a novel isa and on-chip learning,” *National Science Review*, vol. 11, no. 5, p. nwae102, 2024.
- [17] W. Maass, “Networks of spiking neurons: the third generation of neural network models,” *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

- [18] A. Balaji, A. Das, Y. Wu, K. Huynh, F. G. Dell’Anna, G. Indiveri, J. L. Krichmar, N. D. Dutt, S. Schaafsma, and F. Catthoor, “Mapping spiking neural networks to neuromorphic hardware,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 1, pp. 76–86, 2019.
- [19] N. Rathi, I. Chakraborty, A. Kosta, A. Sengupta, A. Ankit, P. Panda, and K. Roy, “Exploring neuromorphic computing based on spiking neural networks: Algorithms to hardware,” *ACM Computing Surveys*, vol. 55, no. 12, pp. 1–49, 2023.
- [20] A. Bittar and P. N. Garner, “A surrogate gradient spiking baseline for speech command recognition,” *Frontiers in Neuroscience*, vol. 16, p. 865897, 2022.
- [21] S. Zhang, Q. Yang, C. Ma, J. Wu, H. Li, and K. C. Tan, “Tc-lif: A two-compartment spiking neuron model for long-term sequential modelling,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 15, 2024, pp. 16 838–16 847.
- [22] S. Park, D. Lee, and S. Yoon, “Noise-robust deep spiking neural networks with temporal information,” in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 373–378.
- [23] G. Ma, R. Yan, and H. Tang, “Exploiting noise as a resource for computation and learning in spiking neural networks,” *Patterns*, vol. 4, no. 10, 2023.
- [24] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, “Spatio-temporal backpropagation for training high-performance spiking neural networks,” *Frontiers in Neuroscience*, vol. 12, p. 331, 2018.
- [25] E. O. Neftci, H. Mostafa, and F. Zenke, “Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks,” *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

- 
- [26] Q. Meng, M. Xiao, S. Yan, Y. Wang, Z. Lin, and Z.-Q. Luo, “Training high-performance low-latency spiking neural networks by differentiation on spike representation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 444–12 453.
- [27] Y. Li, Y. Guo, S. Zhang, S. Deng, Y. Hai, and S. Gu, “Differentiable spike: Rethinking gradient-descent for training spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 23 426–23 439, 2021.
- [28] T. Bu, W. Fang, J. Ding, P. DAI, Z. Yu, and T. Huang, “Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks,” in *International Conference on Learning Representations*, 2022, pp. 1–13.
- [29] Q. Yang, J. Wu, M. Zhang, Y. Chua, X. Wang, and H. Li, “Training spiking neural networks with local tandem learning,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 12 662–12 676, 2022.
- [30] Z. Wang, R. Jiang, S. Lian, R. Yan, and H. Tang, “Adaptive smoothing gradient learning for spiking neural networks,” in *International Conference on Machine Learning*. PMLR, 2023, pp. 35 798–35 816.
- [31] H. Zheng, Y. Wu, L. Deng, Y. Hu, and G. Li, “Going deeper with directly-trained larger spiking neural networks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 11 062–11 070.
- [32] S. Deng, Y. Li, S. Zhang, and S. Gu, “Temporal efficient training of spiking neural network via gradient re-weighting,” *arXiv preprint arXiv:2202.11946*, 2022.
- [33] W. Fang, Z. Yu, Y. Chen, T. Huang, T. Masquelier, and Y. Tian, “Deep residual learning in spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 21 056–21 069, 2021.

- [34] G. Shen, D. Zhao, Y. Dong, and Y. Zeng, “Brain-inspired neural circuit evolution for spiking neural networks,” *Proceedings of the National Academy of Sciences*, vol. 120, no. 39, p. e2218173120, 2023.
- [35] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, “Long short-term memory and learning-to-learn in networks of spiking neurons,” in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018, pp. 795—805.
- [36] M. Yao, X. Qiu, T. Hu, J. Hu, Y. Chou, K. Tian, J. Liao, L. Leng, B. Xu, and G. Li, “Scaling spike-driven transformer with efficient spike firing approximation training,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 04, pp. 2973–2990, 2025.
- [37] Q. Yu, H. Tang, K. C. Tan, and H. Yu, “A brain-inspired spiking neural network model with temporal encoding and learning,” *Neurocomputing*, vol. 138, pp. 3–13, 2014.
- [38] S. Deng and S. Gu, “Optimal conversion of conventional artificial neural networks to spiking neural networks,” *arXiv preprint arXiv:2103.00476*, 2021.
- [39] J. Wu, C. Xu, X. Han, D. Zhou, M. Zhang, H. Li, and K. C. Tan, “Progressive tandem learning for pattern recognition with deep spiking neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7824–7840, 2021.
- [40] Q. Yang, Q. Liu, N. Li, M. Ge, Z. Song, and H. Li, “Svad: A robust, low-power, and light-weight voice activity detection with spiking neural networks,” in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 221–225.

- 
- [41] X. Hao, C. Ma, Q. Yang, K. C. Tan, and J. Wu, “When audio denoising meets spiking neural network,” in *2024 IEEE Conference on Artificial Intelligence (CAI)*. IEEE, 2024, pp. 1524–1527.
- [42] B. Glackin, J. A. Wall, T. M. McGinnity, L. P. Maguire, and L. J. McDaid, “A spiking neural network model of the medial superior olive using spike timing dependent plasticity for sound localization,” *Frontiers in Computational Neuroscience*, vol. 4, p. 18, 2010.
- [43] J. A. Wall, L. J. McDaid, L. P. Maguire, and T. M. McGinnity, “Spiking neural network model of sound localization using the interaural intensity difference,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 4, pp. 574–586, 2012.
- [44] Z. Pan, M. Zhang, J. Wu, J. Wang, and H. Li, “Multi-tone phase coding of interaural time difference for sound source localization with spiking neural networks,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2656–2670, 2021.
- [45] Y. Luo, Q. Fu, J. Xie, Y. Qin, G. Wu, J. Liu, F. Jiang, Y. Cao, and X. Ding, “Eeg-based emotion classification using spiking neural networks,” *IEEE Access*, vol. 8, pp. 46 007–46 016, 2020.
- [46] B. Yin, F. Corradi, and S. M. Bohté, “Accurate and efficient time-domain classification with adaptive spiking recurrent neural networks,” *Nature Machine Intelligence*, vol. 3, no. 10, pp. 905–913, 2021.
- [47] H. Zheng, Z. Zheng, R. Hu, B. Xiao, Y. Wu, F. Yu, X. Liu, G. Li, and L. Deng, “Temporal dendritic heterogeneity incorporated with spiking neural networks for learning multi-timescale dynamics,” *Nature Communications*, vol. 15, no. 1, p. 277, 2024.

- [48] G. Tang, A. Shah, and K. P. Michmizos, “Spiking neural network on neuromorphic hardware for energy-efficient unidimensional slam,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4176–4181.
- [49] J. Ding, B. Dong, F. Heide, Y. Ding, Y. Zhou, B. Yin, and X. Yang, “Biologically inspired dynamic thresholds for spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 6090–6103, 2022.
- [50] T. DeWolf, “Spiking neural networks take control,” *Science Robotics*, vol. 6, no. 58, p. eabk3268, 2021.
- [51] A. N. Burkitt, “A review of the integrate-and-fire neuron model: I. homogeneous synaptic input,” *Biological Cybernetics*, vol. 95, pp. 1–19, 2006.
- [52] J. Gjorgjieva, G. Drion, and E. Marder, “Computational implications of biophysical diversity and multiple timescales in neurons and synapses for circuit performance,” *Current Opinion in Neurobiology*, vol. 37, pp. 44–52, 2016.
- [53] N. Spruston, “Pyramidal neurons: dendritic structure and synaptic integration,” *Nature Reviews Neuroscience*, vol. 9, no. 3, pp. 206–221, 2008.
- [54] W. Rall, “Cable theory for dendritic neurons,” in *Methods in neuronal modeling: From synapses to networks*, 1989, pp. 9–92.
- [55] C. V. Buhusi and W. H. Meck, “What makes us tick? functional and neural mechanisms of interval timing,” *Nature Reviews Neuroscience*, vol. 6, no. 10, pp. 755–765, 2005.
- [56] P. Tacikowski, G. Kalender, D. Ciliberti, and I. Fried, “Human hippocampal and entorhinal neurons encode the temporal structure of experience,” *Nature*, vol. 635, no. 8037, pp. 160–167, 2024.

- 
- [57] H. Sabbella, A. Mukherjee, T. Kandappu, S. Dey, A. Pal, A. Misra, and D. Ma, “The promise of spiking neural networks for ubiquitous computing: A survey and new perspectives,” *arXiv preprint arXiv:2506.01737*, 2025.
- [58] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.
- [59] D. Huh and T. J. Sejnowski, “Gradient descent for spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.
- [60] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943.
- [61] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain.” *Psychological Review*, vol. 65, no. 6, p. 386, 1958.
- [62] J. K. Eshraghian, M. Ward, E. O. Neftci, X. Wang, G. Lenz, G. Dwivedi, M. Bennamoun, D. S. Jeong, and W. D. Lu, “Training spiking neural networks using lessons from deep learning,” *Proceedings of the IEEE*, vol. 111, no. 9, pp. 1016–1054, 2023.
- [63] A. L. Hodgkin and A. F. Huxley, “A quantitative description of membrane current and its application to conduction and excitation in nerve,” *The Journal of Physiology*, vol. 117, no. 4, p. 500, 1952.
- [64] W. Rall, “Theoretical significance of dendritic trees for neuronal input-output relations,” *Neural theory and modeling*, pp. 63–97, 1964.
- [65] E. M. Izhikevich, “Simple model of spiking neurons,” *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

- [66] —, “Resonate-and-fire neurons,” *Neural Networks*, vol. 14, no. 6-7, pp. 883–894, 2001.
- [67] W. Gerstner, R. Ritz, and J. L. Van Hemmen, “Why spikes? hebbian learning and retrieval of time-resolved excitation patterns,” *Biological Cybernetics*, vol. 69, no. 5, pp. 503–515, 1993.
- [68] W. Fang, Z. Yu, Y. Chen, T. Masquelier, T. Huang, and Y. Tian, “Incorporating learnable membrane time constant to enhance learning of spiking neural networks,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2661–2671.
- [69] C. Jiang and Y. Zhang, “Klif: An optimized spiking neuron unit for tuning surrogate gradient slope and membrane potential,” *arXiv preprint arXiv:2302.09238*, 2023.
- [70] X. Yao, F. Li, Z. Mo, and J. Cheng, “GLIF: A unified gated leaky integrate-and-fire neuron for spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 32 160–32 171, 2022.
- [71] A. Shaban, S. S. Bezugam, and M. Suri, “An adaptive threshold neuron for recurrent spiking neural networks with nanodevice hardware implementation,” *Nature Communications*, vol. 12, no. 1, p. 4234, 2021.
- [72] D. Salaj, A. Subramoney, C. Kraisnikovic, G. Bellec, R. Legenstein, and W. Maass, “Spike frequency adaptation supports network computations on temporally dispersed information,” *eLife*, vol. 10, p. e65459, 2021.
- [73] A. Rao, P. Plank, A. Wild, and W. Maass, “A long short-term memory for ai applications in spike-based neuromorphic hardware,” *Nature Machine Intelligence*, vol. 4, no. 5, pp. 467–479, 2022.

- 
- [74] B. Yin, F. Corradi, and S. M. Bohté, “Accurate online training of dynamical spiking neural networks through forward propagation through time,” *Nature Machine Intelligence*, vol. 5, no. 5, pp. 518–527, 2023.
- [75] Y. Huang, X. Lin, H. Ren, H. Fu, Y. Zhou, Z. Liu, B. Pan, and B. Cheng, “Clif: Complementary leaky integrate-and-fire neuron for spiking neural networks,” in *International Conference on Machine Learning*. PMLR, 2024, pp. 19 949–19 972.
- [76] L. Wang and Z. Yu, “Autaptic synaptic circuit enhances spatio-temporal predictive learning of spiking neural networks,” in *International Conference on Machine Learning*. PMLR, 2024, pp. 52 083–52 098.
- [77] I. Hammouamri, I. Khalfaoui-Hassani, and T. Masquelier, “Learning delays in spiking neural networks using dilated convolutions with learnable spacings,” in *International Conference on Learning Representations*, 2024.
- [78] P. Sun, J. Wu, M. Zhang, P. Devos, and D. Botteldooren, “Delay learning based on temporal coding in spiking neural networks,” *Neural Networks*, vol. 180, p. 106678, 2024.
- [79] W. Fang, Z. Yu, Z. Zhou, D. Chen, Y. Chen, Z. Ma, T. Masquelier, and Y. Tian, “Parallel spiking neurons with high efficiency and ability to learn long-term dependencies,” in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 53 674–53 687.
- [80] X. Luo, M. Yao, Y. Chou, B. Xu, and G. Li, “Integer-valued training and spike-driven inference spiking neural network for high-performance and energy-efficient object detection,” in *European Conference on Computer Vision*. Springer, 2024, pp. 253–272.

- [81] W. Guo, M. E. Fouda, A. M. Eltawil, and K. N. Salama, “Neural coding in spiking neural networks: A comparative study for robust neuromorphic systems,” *Frontiers in Neuroscience*, vol. 15, p. 638474, 2021.
- [82] E. Adrian and Y. Zotterman, “The impulses produced by sensory nerve endings: Part 3. impulses set up by touch and pressure,” *The Journal of Physiology*, vol. 61, no. 4, pp. 465–483, 1926.
- [83] K. H. Srivastava, C. M. Holmes, M. Vellema, A. R. Pack, C. P. Elemans, I. Nemenman, and S. J. Sober, “Motor control by precisely timed spike patterns,” *Proceedings of the National Academy of Sciences*, vol. 114, no. 5, pp. 1171–1176, 2017.
- [84] F. Rieke, D. Warland, R. d. R. Van Steveninck, and W. Bialek, *Spikes: exploring the neural code*. MIT press, 1999.
- [85] S. Thorpe, D. Fize, and C. Marlot, “Speed of processing in the human visual system,” *Nature*, vol. 381, no. 6582, pp. 520–522, 1996.
- [86] A. Carleton, R. Accolla, and S. A. Simon, “Coding in the mammalian gustatory system,” *Trends in Neurosciences*, vol. 33, no. 7, pp. 326–334, 2010.
- [87] B. Fontaine and H. Peremans, “Bat echolocation processing using first-spike latency coding,” *Neural Networks*, vol. 22, no. 10, pp. 1372–1382, 2009.
- [88] T. Gollisch and M. Meister, “Rapid neural coding in the retina with relative spike latencies,” *Science*, vol. 319, no. 5866, pp. 1108–1111, 2008.
- [89] G. Buzsaki and A. Draguhn, “Neuronal oscillations in cortical networks,” *Science*, vol. 304, no. 5679, pp. 1926–1929, 2004.
- [90] J. O’Keefe and M. L. Recce, “Phase relationship between hippocampal place units and the eeg theta rhythm,” *Hippocampus*, vol. 3, no. 3, pp. 317–330, 1993.

- 
- [91] M. A. Montemurro, M. J. Rasch, Y. Murayama, N. K. Logothetis, and S. Panzeri, “Phase-of-firing coding of natural visual stimuli in primary visual cortex,” *Current Biology*, vol. 18, no. 5, pp. 375–380, 2008.
- [92] N. Imam and T. A. Cleland, “Rapid online learning and robust recall in a neuromorphic olfactory circuit,” *Nature Machine Intelligence*, vol. 2, no. 3, pp. 181–191, 2020.
- [93] F. Zeldenrust, W. J. Wadman, and B. Englitz, “Neural coding with bursts—current state and future perspectives,” *Frontiers in Computational Neuroscience*, vol. 12, p. 48, 2018.
- [94] J. E. Lisman, “Bursts as a unit of neural information: making unreliable synapses reliable,” *Trends in Neurosciences*, vol. 20, no. 1, pp. 38–43, 1997.
- [95] S. Park, S. Kim, B. Na, and S. Yoon, “T2fsnn: Deep spiking neural networks with time-to-first-spike coding,” in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.
- [96] M. Zhang, J. Wang, J. Wu, A. Belatreche, B. Amornpaisannon, Z. Zhang, V. P. K. Miriyala, H. Qu, Y. Chua, T. E. Carlson *et al.*, “Rectified linear postsynaptic potential function for backpropagation in deep spiking neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 5, pp. 1947–1958, 2021.
- [97] L. Zhang, S. Zhou, T. Zhi, Z. Du, and Y. Chen, “Tdsnn: From deep neural networks to deep spike neural networks with temporal-coding,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1319–1326.
- [98] Y. Chen, H. Qu, M. Zhang, and Y. Wang, “Deep spiking neural network with neural oscillation and spike-phase information,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 7073–7080.

- [99] S. Park, S. Kim, H. Choe, and S. Yoon, “Fast and efficient information transmission with burst spikes in deep spiking neural networks,” in *2019 56th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2019, pp. 1–6.
- [100] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.
- [101] W. Gerstner, R. Kempter, J. L. Van Hemmen, and H. Wagner, “A neuronal learning rule for sub-millisecond temporal coding,” *Nature*, vol. 383, no. 6595, pp. 76–78, 1996.
- [102] N. Caporale and Y. Dan, “Spike timing–dependent plasticity: a hebbian learning rule,” *Annual Review of Neuroscience*, vol. 31, no. 1, pp. 25–46, 2008.
- [103] T. Bu, J. Ding, Z. Yu, and T. Huang, “Optimized potential initialization for low-latency spiking neural networks,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 1, 2022, pp. 11–20.
- [104] Y. Cao, Y. Chen, and D. Khosla, “Spiking deep convolutional neural networks for energy-efficient object recognition,” *International Journal of Computer Vision*, vol. 113, pp. 54–66, 2015.
- [105] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, “Going deeper in spiking neural networks: Vgg and residual architectures,” *Frontiers in Neuroscience*, vol. 13, p. 95, 2019.
- [106] B. Han, G. Srinivasan, and K. Roy, “Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 558–13 567.
- [107] N. Rathi, G. Srinivasan, P. Panda, and K. Roy, “Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation,” *arXiv preprint arXiv:2005.01807*, 2020.

- 
- [108] Y. Li, S. Deng, X. Dong, R. Gong, and S. Gu, “A free lunch from ann: Towards efficient, accurate spiking neural networks calibration,” in *International Conference on Machine Learning*. PMLR, 2021, pp. 6316–6325.
  - [109] Y. Wu, L. Deng, G. Li, J. Zhu, Y. Xie, and L. Shi, “Direct training for spiking neural networks: Faster, larger, better,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1311–1318.
  - [110] S. B. Shrestha and G. Orchard, “Slayer: Spike layer error reassignment in time,” *Advances in Neural Information Processing Systems*, vol. 31, 2018.
  - [111] Q. Xu, J. Shen, X. Ran, H. Tang, G. Pan, and J. K. Liu, “Robust transcoding sensory information with neural spikes,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 5, pp. 1935–1946, 2021.
  - [112] C. Duan, J. Ding, S. Chen, Z. Yu, and T. Huang, “Temporal effective batch normalization in spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 34 377–34 390, 2022.
  - [113] Y. Guo, Y. Chen, L. Zhang, X. Liu, Y. Wang, X. Huang, and Z. Ma, “Im-loss: information maximization loss for spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 156–166, 2022.
  - [114] Y. Hu, L. Deng, Y. Wu, M. Yao, and G. Li, “Advancing spiking neural networks toward deep residual learning,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 2, pp. 2353–2367, 2024.
  - [115] G. Wang, Y. Sun, S. Cheng, and S. Song, “Evolving connectivity for recurrent spiking neural networks,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 2991–3007, 2023.
  - [116] L. Qin, Z. Wang, R. Yan, and H. Tang, “Attention-based deep spiking neural networks for temporal credit assignment problems,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 8, pp. 10 301–10 311, 2023.

- [117] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [118] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” University of Toronto, Tech. Rep., 2009.
- [119] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255.
- [120] J. Wu, Q. Zhang, and G. Xu, “Tiny imagenet challenge,” *Technical Report*, 2017.
- [121] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, “Converting static image datasets to spiking neuromorphic datasets using saccades,” *Frontiers in Neuroscience*, vol. 9, p. 437, 2015.
- [122] H. Li, H. Liu, X. Ji, G. Li, and L. Shi, “Cifar10-dvs: an event-stream dataset for object classification,” *Frontiers in Neuroscience*, vol. 11, p. 244131, 2017.
- [123] R. Gopalakrishnan, Y. Chua, and L. R. Iyer, “Classifying neuromorphic data using a deep learning framework for image classification,” in *International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 2018, pp. 1520–1524.
- [124] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories,” in *Conference on Computer Vision and Pattern Recognition Workshop*. IEEE, 2004, pp. 178–178.
- [125] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, “A low power, fully event-

- based gesture recognition system,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.
- [126] Y. Wang, X. Zhang, Y. Shen, B. Du, G. Zhao, L. Cui, and H. Wen, “Event-stream representation for human gaits identification using deep neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3436–3449, 2021.
- [127] S. Zhou, B. Yang, M. Yuan, R. Jiang, R. Yan, G. Pan, and H. Tang, “Enhancing SNN-based spatio-temporal learning: A benchmark dataset and cross-modality attention model,” *Neural Networks*, vol. 180, p. 106677, 2024.
- [128] X. Wang, Z. Wu, B. Jiang, Z. Bao, L. Zhu, G. Li, Y. Wang, and Y. Tian, “Hardvs: Revisiting human activity recognition with dynamic vision sensors,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 6, 2024, pp. 5615–5623.
- [129] P. Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” *arXiv preprint arXiv:1804.03209*, 2018.
- [130] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, and D. S. Pallett, “Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1,” *NASA STI/Recon technical report n*, vol. 93, p. 27403, 1993.
- [131] B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke, “The heidelberg spiking data sets for the systematic evaluation of spiking neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 7, pp. 2744–2757, 2020.
- [132] J. Wu, Y. Chua, M. Zhang, G. Li, H. Li, and K. C. Tan, “A tandem learning rule for effective training and rapid inference of deep spiking neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 1, pp. 446–460, 2021.

- [133] J. Wu, Y. Chua, M. Zhang, H. Li, and K. C. Tan, “A spiking neural network framework for robust sound classification,” *Frontiers in Neuroscience*, vol. 12, p. 836, 2018.
- [134] J. Wu, E. Yilmaz, M. Zhang, H. Li, and K. C. Tan, “Deep spiking neural networks for large vocabulary automatic speech recognition,” *Frontiers in Neuroscience*, vol. 14, p. 199, 2020.
- [135] S. Thorpe, A. Delorme, and R. Van Rullen, “Spike-based strategies for rapid processing,” *Neural Networks*, vol. 14, no. 6-7, pp. 715–725, 2001.
- [136] J. Ding, Z. Yu, Y. Tian, and T. Huang, “Optimal ann-snn conversion for fast and accurate inference in deep spiking neural networks,” in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. International Joint Conferences on Artificial Intelligence Organization, 2021, pp. 2328–2336.
- [137] Z. Pan, J. Wu, M. Zhang, H. Li, and Y. Chua, “Neural population coding for effective temporal classification,” in *2019 International Joint Conference on Neural Networks (IJCNN)*, 2019, pp. 1–8.
- [138] C. Kayser, M. A. Montemurro, N. K. Logothetis, and S. Panzeri, “Spike-phase coding boosts and stabilizes information carried by spatial and temporal spike patterns,” *Neuron*, vol. 61, no. 4, pp. 597–608, 2009.
- [139] J. Kim, H. Kim, S. Huh, J. Lee, and K. Choi, “Deep neural networks with weighted spikes,” *Neurocomputing*, vol. 311, pp. 373–386, 2018.
- [140] E. M. Izhikevich, N. S. Desai, E. C. Walcott, and F. C. Hoppensteadt, “Bursts as a unit of neural information: selective communication via resonance,” *Trends in Neurosciences*, vol. 26, no. 3, pp. 161–167, 2003.

- 
- [141] Y. Li and Y. Zeng, “Efficient and accurate conversion of spiking neural network with burst spikes,” in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, L. D. Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization, 2022, pp. 2485–2491.
- [142] B. Rueckauer and S.-C. Liu, “Conversion of analog to spiking neural networks using sparse temporal coding,” in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5.
- [143] S. Oh, D. Kwon, G. Yeom, W.-M. Kang, S. Lee, S. Y. Woo, J. Kim, and J.-H. Lee, “Neuron circuits for low-power spiking neural networks using time-to-first-spike encoding,” *IEEE Access*, vol. 10, pp. 24 444–24 455, 2022.
- [144] G. Buzsáki, “How do neurons sense a spike burst?” *Neuron*, vol. 73, no. 5, pp. 857–859, 2012.
- [145] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, “Conversion of continuous-valued deep networks to efficient event-driven networks for image classification,” *Frontiers in Neuroscience*, vol. 11, p. 682, 2017.
- [146] R. Gütiğ and H. Sompolinsky, “The tempotron: a neuron that learns spike timing-based decisions,” *Nature Neuroscience*, vol. 9, no. 3, pp. 420–428, 2006.
- [147] L. A. Jeffress, “A place theory of sound localization.” *Journal of Comparative and Physiological Psychology*, vol. 41, no. 1, p. 35, 1948.
- [148] D. F. Goodman and R. Brette, “Spike-timing-based computation in sound localization,” *PLoS Computational Biology*, vol. 6, no. 11, p. e1000993, 2010.
- [149] Y. Bengio, N. Léonard, and A. Courville, “Estimating or propagating gradients through stochastic neurons for conditional computation,” *arXiv preprint arXiv:1308.3432*, 2013.

- [150] W. Li, P.-C. Chu, G.-Z. Liu, Y.-B. Tian, T.-H. Qiu, and S.-M. Wang, “An image classification algorithm based on hybrid quantum classical convolutional neural network,” *Quantum Engineering*, vol. 2022, no. 1, p. 5701479, 2022.
- [151] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations*, 2015.
- [152] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 770–778.
- [153] X. Qian, B. Sharma, A. E. Abrid, and H. Li, “Slocas: A database for joint sound localization and classification,” *arXiv preprint arXiv:2108.02539*, 2021.
- [154] B. Han and K. Roy, “Deep spiking neural network: Energy efficiency through time based coding,” in *European Conference on Computer Vision*. Springer, 2020, pp. 388–404.
- [155] N. Rathi and K. Roy, “Diet-snn: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 6, pp. 3174–3182, 2021.
- [156] S. Kundu, G. Datta, M. Pedram, and P. A. Beerel, “Spike-thrift: Towards energy-efficient deep spiking neural networks by limiting spiking activity via attention-guided compression,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. IEEE, 2021, pp. 3953–3962.
- [157] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura *et al.*, “A million spiking-neuron integrated circuit with a scalable communication network and interface,” *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

- 
- [158] M. Yao, G. Zhao, H. Zhang, Y. Hu, L. Deng, Y. Tian, B. Xu, and G. Li, “Attention spiking neural networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 9393–9410, 2023.
- [159] S. Han, J. Pool, J. Tran, and W. J. Dally, “Learning both weights and connections for efficient neural networks,” in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. MIT Press, 2015, p. 1135–1143.
- [160] W. He, P. Motlicek, and J.-M. Odobez, “Joint Localization and Classification of Multiple Sound Sources Using a Multi-task Neural Network,” in *Proceedings of INTERSPEECH*, 2018, pp. 312–316.
- [161] S. Adavanne, A. Politis, J. Nikunen, and T. Virtanen, “Sound event localization and detection of overlapping sources using convolutional recurrent neural networks,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 13, no. 1, pp. 34–48, 2018.
- [162] Y. Cao, T. Iqbal, Q. Kong, F. An, W. Wang, and M. D. Plumbley, “An improved event-independent network for polyphonic sound event localization and detection,” in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 885–889.
- [163] B. Yang, H. Liu, and X. Li, “Srp-dnn: Learning direct-path phase difference for multiple moving sound source localization,” in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 721–725.
- [164] Y. Wang, B. Yang, and X. Li, “Fn-ssl: Full-band and narrow-band fusion for sound source localization,” in *Proceedings of INTERSPEECH*, 2023.

- [165] G. Bellec, F. Scherr, A. Subramoney, E. Hajek, D. Salaj, R. Legenstein, and W. Maass, “A solution to the learning dilemma for recurrent networks of spiking neurons,” *Nature Communications*, vol. 11, no. 1, p. 3625, 2020.
- [166] M. Hines, “Efficient computation of branched nerve equations,” *International Journal of Bio-medical Computing*, vol. 15, no. 1, pp. 69–76, 1984.
- [167] M. E. Larkum, J. J. Zhu, and B. Sakmann, “A new cellular mechanism for coupling inputs arriving at different cortical layers,” *Nature*, vol. 398, no. 6725, pp. 338–341, 1999.
- [168] A. Gidon, T. A. Zolnik, P. Fidzinski, F. Bolduan, A. Papoutsi, P. Poirazi, M. Holtkamp, I. Vida, and M. E. Larkum, “Dendritic action potentials and computation in human layer 2/3 cortical neurons,” *Science*, vol. 367, no. 6473, pp. 83–87, 2020.
- [169] G. Major, M. E. Larkum, and J. Schiller, “Active properties of neocortical pyramidal neuron dendrites,” *Annual Review of Neuroscience*, vol. 36, pp. 1–24, 2013.
- [170] J. Acharya, A. Basu, R. Legenstein, T. Limbacher, P. Poirazi, and X. Wu, “Dendritic computing: branching deeper into machine learning,” *Neuroscience*, vol. 489, pp. 275–289, 2022.
- [171] E. Martin and C. Cundy, “Parallelizing linear recurrent neural nets over sequence length,” in *International Conference on Learning Representations*, 2018, pp. 1–9.
- [172] A. Gu, K. Goel, and C. Re, “Efficiently modeling long sequences with structured state spaces,” in *International Conference on Learning Representations*, 2022, pp. 1–15.

- 
- [173] J. T. Smith, A. Warrington, and S. Linderman, “Simplified state space layers for sequence modeling,” in *The Eleventh International Conference on Learning Representations*, 2023, pp. 1–13.
- [174] G. Buzsáki and X.-J. Wang, “Mechanisms of gamma oscillations,” *Annual Review of Neuroscience*, vol. 35, pp. 203–225, 2012.
- [175] R. D. Traub, R. K. Wong, R. Miles, and H. Michelson, “A model of a ca3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances,” *Journal of Neurophysiology*, vol. 66, no. 2, pp. 635–650, 1991.
- [176] P. F. Pinsky and J. Rinzel, “Intrinsic and network rhythmogenesis in a reduced traub model for ca3 neurons,” *Journal of Computational Neuroscience*, vol. 1, pp. 39–60, 1994.
- [177] R. A. DeCarlo, *Linear systems: A state variable approach with numerical implementation*. Prentice-Hall, Inc., 1989.
- [178] M. Davies *et al.*, “Taking neuromorphic computing to the next level with loihi2,” *Intel Labs’ Loihi*, vol. 2, pp. 1–7, 2021.
- [179] M. Harris, S. Sengupta, and J. D. Owens, “Parallel prefix sum (scan) with cuda,” *GPU gems*, vol. 3, no. 39, pp. 851–876, 2007.
- [180] S. Higuchi, S. Kairat, S. M. Bohte, and S. Otte, “Balanced resonate-and-fire neurons,” 2024.
- [181] M. Yao, J. Hu, Z. Zhou, L. Yuan, Y. Tian, B. Xu, and G. Li, “Spike-driven transformer,” *Advances in Neural Information Processing Systems*, vol. 36, 2024.
- [182] M.-I. Stan and O. Rhodes, “Learning long sequences in spiking neural networks,” *Scientific Reports*, vol. 14, no. 1, p. 21957, 2024.

- [183] T. E. Huber, J. Lecomte, B. Polovnikov, and A. von Arnim, “Scaling up resonate-and-fire networks for fast deep learning,” in *European Conference on Computer Vision*. Springer, 2024, pp. 241–258.
- [184] P. Sun, E. Eqlimi, Y. Chua, P. Devos, and D. Botteldooren, “Adaptive axonal delays in feedforward spiking neural networks for accurate spoken word recognition,” in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.
- [185] L. Deckers, L. Van Damme, W. Van Leekwijck, I. J. Tsang, and S. Latré, “Co-learning synaptic delays, weights and adaptation in spiking neural networks,” *Frontiers in Neuroscience*, vol. 18, p. 1360300, 2024.
- [186] P. G. Malettira, S. Negi, W. Ponghiran, and K. Roy, “TSkips: Efficiency through explicit temporal delay connections in spiking neural networks,” *Transactions on Machine Learning Research*, 2025.
- [187] F. Zenke and T. P. Vogels, “The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks,” *Neural Computation*, vol. 33, no. 4, pp. 899–925, 2021.
- [188] C. Zhou, L. Yu, Z. Zhou, H. Zhang, Z. Ma, H. Zhou, and Y. Tian, “Spikingformer: Spike-driven residual learning for transformer-based spiking neural network,” *arXiv preprint arXiv:2304.11954*, 2023. [Online]. Available: <https://arxiv.org/abs/2304.11954>
- [189] Q. V. Le, N. Jaitly, and G. E. Hinton, “A simple way to initialize recurrent networks of rectified linear units,” *arXiv preprint arXiv:1504.00941*, 2015.
- [190] Z. Zhou, Y. Zhu, C. He, Y. Wang, S. YAN, Y. Tian, and L. Yuan, “Spikformer: When spiking neural network meets transformer,” in *International Conference on Learning Representations*, 2023.

- 
- [191] Y. Guo, Y. Zhang, Y. Chen, W. Peng, X. Liu, L. Zhang, X. Huang, and Z. Ma, “Membrane potential batch normalization for spiking neural networks,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023, pp. 19 420–19 430.
- [192] Y. Guo, Y. Chen, L. Zhang, X. Liu, X. Tong, Y. Ou, X. Huang, and Z. Ma, “Inflor-snn: Reducing information loss for spiking neural networks,” *arXiv preprint arXiv:2307.04356*, 2023.
- [193] H. Jaeger, D. Doorakkers, C. Lawrence, and G. Indiveri, “Dimensions of timescales in neuromorphic computing systems,” *arXiv preprint arXiv:2102.10648*, 2021.
- [194] M. Horowitz, “1.1 computing’s energy problem (and what we can do about it),” in *IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*. IEEE, 2014, pp. 10–14.
- [195] A. C. Schapiro, T. T. Rogers, N. I. Cordova, N. B. Turk-Browne, and M. M. Botvinick, “Neural representations of events arise from temporal community structure,” *Nature Neuroscience*, vol. 16, no. 4, pp. 486–492, 2013.
- [196] J. J. Paton and D. V. Buonomano, “The neural basis of timing: distributed mechanisms for diverse functions,” *Neuron*, vol. 98, no. 4, pp. 687–705, 2018.
- [197] D. B. Omer, L. Las, and N. Ulanovsky, “Contextual and pure time coding for self and other in the hippocampus,” *Nature Neuroscience*, vol. 26, no. 2, pp. 285–294, 2023.
- [198] C. A. Czeisler, J. F. Duffy, T. L. Shanahan, E. N. Brown, J. F. Mitchell, D. W. Rimmer, J. M. Ronda, E. J. Silva, J. S. Allan, J. S. Emens *et al.*, “Stability, precision, and near-24-hour period of the human circadian pacemaker,” *Science*, vol. 284, no. 5423, pp. 2177–2181, 1999.

- [199] M. Tang, L.-H. Cao, T. Yang, S.-X. Ma, B.-Y. Jing, N. Xiao, S. Xu, K.-R. Leng, D. Yang, M.-T. Li *et al.*, “An extra-clock ultradian brain oscillator sustains circadian timekeeping,” *Science Advances*, vol. 8, no. 35, p. eabo5506, 2022.
- [200] C. I. De Zeeuw, F. E. Hoebeek, and M. Schonewille, “Causes and consequences of oscillations in the cerebellar cortex,” *Neuron*, vol. 58, no. 5, pp. 655–658, 2008.
- [201] R. Llinás, “Inferior olive oscillation as the temporal basis for motricity and oscillatory reset as the basis for motor error correction,” *Neuroscience*, vol. 162, no. 3, pp. 797–804, 2009.
- [202] H. Eichenbaum, “Time cells in the hippocampus: a new dimension for mapping memories,” *Nature Reviews Neuroscience*, vol. 15, no. 11, pp. 732–744, 2014.
- [203] G. Buzsáki and D. Tingley, “Space and time: the hippocampus as a sequence generator,” *Trends in Cognitive Sciences*, vol. 22, no. 10, pp. 853–869, 2018.
- [204] M. S. Matell, W. H. Meck, and M. A. Nicolelis, “Interval timing and the encoding of signal duration by ensembles of cortical and striatal neurons.” *Behavioral Neuroscience*, vol. 117, no. 4, p. 760, 2003.
- [205] G. B. Mello, S. Soares, and J. J. Paton, “A scalable population code for time in the striatum,” *Current Biology*, vol. 25, no. 9, pp. 1113–1122, 2015.
- [206] C. Ma, X. Chen, Y. Li, Q. Yang, Y. Wu, G. Li, G. Pan, H. Tang, K. C. Tan, and J. Wu, “Spiking neural networks for temporal processing: Status quo and future prospects,” *arXiv preprint arXiv:2502.09449*, 2025.
- [207] B. Hutcheon and Y. Yarom, “Resonance, oscillation and the intrinsic frequency preferences of neurons,” *Trends in Neurosciences*, vol. 23, no. 5, pp. 216–222, 2000.

- 
- [208] R. Zemankovics, S. Káli, O. Paulsen, T. F. Freund, and N. Hájos, “Differences in subthreshold resonance of hippocampal pyramidal cells and interneurons: the role of h-current and passive membrane characteristics,” *The Journal of Physiology*, vol. 588, no. 12, pp. 2109–2132, 2010.
- [209] S. M. Sunkin, L. Ng, C. Lau, T. Dolbeare, T. L. Gilbert, C. L. Thompson, M. Hawrylycz, and C. Dang, “Allen brain atlas: an integrated spatio-temporal portal for exploring the central nervous system,” *Nucleic Acids Research*, vol. 41, no. D1, pp. D996–D1008, 2012.
- [210] J. T. Coull and A. Nobre, “Dissociating explicit timing from temporal expectation with fmri,” *Current Opinion in Neurobiology*, vol. 18, no. 2, pp. 137–144, 2008.
- [211] H. Merchant, W. Zarco, O. Pérez, L. Prado, and R. Bartolo, “Measuring time with different neural chronometers during a synchronization-continuation task,” *Proceedings of the National Academy of Sciences*, vol. 108, no. 49, pp. 19 784–19 789, 2011.
- [212] P. Sun, L. Zhu, and D. Botteldooren, “Axonal delay as a short-term memory for feed forward deep spiking neural networks,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 8932–8936.
- [213] J. I. Gold and M. N. Shadlen, “The neural basis of decision making,” *Annu. Rev. Neurosci.*, vol. 30, no. 1, pp. 535–574, 2007.
- [214] R. Ratcliff and G. McKoon, “The diffusion decision model: theory and data for two-choice decision tasks,” *Neural Computation*, vol. 20, no. 4, pp. 873–922, 2008.

- [215] A. D. Ekstrom, M. J. Kahana, J. B. Caplan, T. A. Fields, E. A. Isham, E. L. Newman, and I. Fried, “Cellular networks underlying human spatial navigation,” *Nature*, vol. 425, no. 6954, pp. 184–188, 2003.
- [216] N. Axmacher, F. Mormann, G. Fernández, C. E. Elger, and J. Fell, “Memory formation by neuronal synchronization,” *Brain Research Reviews*, vol. 52, no. 1, pp. 170–182, 2006.
- [217] C. J. MacDonald, K. Q. Lepage, U. T. Eden, and H. Eichenbaum, “Hippocampal “time cells” bridge the gap in memory for discontiguous events,” *Neuron*, vol. 71, no. 4, pp. 737–749, 2011.
- [218] H. Li, Z. Xu, G. Taylor, C. Studer, and T. Goldstein, “Visualizing the loss landscape of neural nets,” in *Neural Information Processing Systems*, 2018.
- [219] M. Arjovsky, A. Shah, and Y. Bengio, “Unitary evolution recurrent neural networks,” in *International Conference on Machine Learning*, vol. 48. PMLR, 2016, pp. 1120–1128.
- [220] D. Auge, J. Hille, F. Kreutz, E. Mueller, and A. Knoll, “End-to-end spiking neural network for speech recognition using resonating input neurons,” in *International Conference on Artificial Neural Networks*. Springer, 2021, pp. 245–256.
- [221] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, “Building a large annotated corpus of english: The penn treebank,” *Computational Linguistics*, vol. 19, no. 2, pp. 313–330, 1993.
- [222] R. Krahe and F. Gabbiani, “Burst firing in sensory systems,” *Nature Reviews Neuroscience*, vol. 5, no. 1, pp. 13–23, 2004.
- [223] S. Moradi, N. Qiao, F. Stefanini, and G. Indiveri, “A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic

- asynchronous processors (dynaps),” *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 1, pp. 106–122, 2017.
- [224] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, “Openai gym,” *arXiv preprint arXiv:1606.01540*, 2016.
- [225] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [226] C. P. Billimoria, R. A. DiCaprio, J. T. Birmingham, L. Abbott, and E. Marder, “Neuromodulation of spike-timing precision in sensory neurons,” *Journal of Neuroscience*, vol. 26, no. 22, pp. 5910–5919, 2006.
- [227] M. O’Kelly, H. Zheng, D. Karthik, and R. Mangharam, “F1tenth: An open-source evaluation environment for continuous control and reinforcement learning,” in *NeurIPS 2019 Competition and Demonstration Track*. PMLR, 2020, pp. 77–89.
- [228] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, “Autonomous vehicles on the edge: A survey on autonomous vehicle racing,” *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 458–488, 2022.
- [229] X. Chen, J. Wu, C. Ma, Y. Yan, Y. Wu, and K. C. Tan, “Pmsn: A parallel multi-compartment spiking neuron for multi-scale temporal processing,” *arXiv preprint arXiv:2408.14917*, 2024.
- [230] H. G. Eyherabide, A. Rokem, A. V. Herz, and I. Samengo, “Burst firing is a neural code in an insect auditory system,” *Frontiers in Computational Neuroscience*, vol. 2, p. 172, 2008.
- [231] K. D. Harris, H. Hirase, X. Leinekugel, D. A. Henze, and G. Buzsáki, “Temporal interaction between single spikes and complex spike bursts in hippocampal pyramidal cells,” *Neuron*, vol. 32, no. 1, pp. 141–149, 2001.

- [232] J. Büchel, D. Zendrikov, S. Solinas, G. Indiveri, and D. R. Muir, “Supervised training of spiking neural networks for robust deployment on mixed-signal neuromorphic processors,” *Scientific Reports*, vol. 11, no. 1, p. 23376, 2021.
- [233] D. Katare, D. Perino, J. Nurmi, M. Warnier, M. Janssen, and A. Y. Ding, “A survey on approximate edge ai for energy efficient autonomous driving services,” *IEEE Communications Surveys & Tutorials*, vol. 25, no. 4, pp. 2714–2754, 2023.
- [234] Z. Yang, T. Wang, Y. Lin, Y. Chen, H. Zeng, J. Pei, J. Wang, X. Liu, Y. Zhou, J. Zhang *et al.*, “A vision chip with complementary pathways for open-world sensing,” *Nature*, vol. 629, no. 8014, pp. 1027–1033, 2024.
- [235] L. He, Y. Xu, W. He, Y. Lin, Y. Tian, Y. Wu, W. Wang, Z. Zhang, J. Han, Y. Tian *et al.*, “Network model with internal complexity bridges artificial intelligence and neuroscience,” *Nature Computational Science*, vol. 4, no. 8, pp. 584–599, 2024.
- [236] P. Sun, J. Wu, M. Zhang, P. Devos, and D. Botteldooren, “Delayed memory unit: Modeling temporal dependency through delay gate,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 36, no. 6, pp. 10 808–10 818, 2025.
- [237] G. M. Weiss, “Wisdsm smartphone and smartwatch activity and biometrics dataset,” *UCI Machine Learning Repository: WISDM Smartphone and Smartwatch Activity and Biometrics Dataset Data Set*, vol. 7, no. 133190-133202, p. 5, 2019.
- [238] M.-H. Lee, O.-Y. Kwon, Y.-J. Kim, H.-K. Kim, Y.-E. Lee, J. Williamson, S. Fazli, and S.-W. Lee, “Eeg dataset and openbmi toolbox for three bci paradigms: an investigation into bci illiteracy,” *GigaScience*, vol. 8, no. 5, p. giz002, 2019.
- [239] G. Tan, Y. Wang, H. Han, Y. Cao, F. Wu, and Z.-J. Zha, “Multi-grained spatio-temporal features perceived network for event-based lip-reading,” in *Proceedings*

- of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 20 094–20 103.
- [240] J. Timcheck, S. B. Shrestha, D. B. D. Rubin, A. Kupryjanow, G. Orchard, L. Pindor, T. Shea, and M. Davies, “The intel neuromorphic dns challenge,” *Neuromorphic Computing and Engineering*, vol. 3, no. 3, p. 034005, 2023.
- [241] X. Hao, C. Ma, Q. Yang, J. Wu, and K. C. Tan, “Towards ultra-low-power neuromorphic speech enhancement with spiking-fullsubnet,” *arXiv preprint arXiv:2410.04785*, 2024.
- [242] H. Bu, J. Du, X. Na, B. Wu, and H. Zheng, “Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline,” in *Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA)*. IEEE, 2017, pp. 1–5.
- [243] M.-P. Hosseini, A. Hosseini, and K. Ahi, “A review on machine learning for eeg signal processing in bioengineering,” *IEEE Reviews in Biomedical Engineering*, vol. 14, pp. 204–218, 2020.
- [244] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, “Espnet: End-to-end speech processing toolkit,” *arXiv preprint arXiv:1804.00015*, 2018.
- [245] X. Chen, J. Wu, H. Tang, Q. Ren, and K. C. Tan, “Unleashing the potential of spiking neural networks for sequential modeling with contextual embedding,” *arXiv preprint arXiv:2308.15150*, 2023.
- [246] S. Bai, J. Z. Kolter, and V. Koltun, “An empirical evaluation of generic convolutional and recurrent networks for sequence modeling,” *arXiv preprint arXiv:1803.01271*, 2018.

- [247] J. Yik, K. Van den Berghe, D. den Blanken, Y. Bouhadjar, M. Fabre, P. Hueber, W. Ke, M. A. Khoei, D. Kleyko, N. Pacik-Nelson *et al.*, “The neurobench framework for benchmarking neuromorphic computing algorithms and systems,” *Nature Communications*, vol. 16, no. 1, p. 1545, 2025.