Selective Plasticity in Spiking Neural Networks for Energy-Efficient Unsupervised On-Chip Learning: Toward Self-Adapting Implantable Neuromorphic Biosignal Processors

Dariush Valipourhafshejani

A Thesis Submitted to the Faculty of Graduate Studies

In Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science

Graduate Program in Electrical Engineering and Computer Science

York University

Toronto, Ontario

September 2026

© Dariush Valipourhafshejani, 2026

Abstract

Self-adapting implantable biosignal processors motivate energy-efficient on-chip learning, but this thesis evaluates the underlying SNN methods on MNIST rather than patient data. An STDP network is decomposed into synaptic weights, adaptive thresholds, and firing-rate statistics; freeze ablations identify their timescales and motivate confidence- and neuron-selectivity-gated updates. Per-neuron selectivity and maturity are then used to diagnose severe class imbalance, derive four local ratio-agnostic plasticity mechanisms, and organize neurons topographically for window-restricted inference. On controlled rare-event MNIST tasks, selective adaptation reduces weight-commit traffic by approximately 80% during adaptation and 92% at steady state. At 1:1000 imbalance, sensitivity rises from 41% to 72%; windowing retains 68% sensitivity while reducing modeled energy from 6.7 to 1.8μJ. A fixed-point FPGA baseline with on-chip STDP reaches 86.2% MNIST accuracy, 131μs latency, and 22μJ measured programmable-logic dynamic energy per inference. These results support selective plasticity toward implantable neuromorphic learning systems.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Dr. Hossein Kassiri, for his guidance, support, and encouragement throughout my graduate studies. His technical insight and thoughtful feedback were invaluable in shaping this research and my development as a researcher.

I am also grateful to the members of my supervisory and examination committee, Dr. Hossein Kassiri, Dr. Amirali Amirsoleimani, and Dr. Kohitij Kar for their time, constructive comments, and valuable suggestions. I would like to thank the members of our research group Mina Sayedi, Abdul Muneeb, Abdelrahman Soliman, Samira Nabavi, Maxim Korosteliov, Demetrios Koudoumnakis, Dhruv Patel, and Ahmed Ashraf for the helpful discussions, collaboration, and support that made this journey both productive and enjoyable. I am especially thankful to Yousef Khazaei for his guidance and collaboration, and to Mohsen Namavar for his guidance and help throughout this work. Finally, I would like to thank my friends, Reza Rahimi, Ali Eskandari, Abolfazl Anvari, and many others, for their constant support and encouragement.

I gratefully acknowledge the financial support of the Connected Minds program at York University through the Connected Minds Master's Scholarship, funded by the Canada First Research Excellence Fund (CFREF). Finally, I am deeply thankful to my family for their patience, encouragement, and unwavering support throughout this journey.

Generative artificial intelligence tools, including OpenAI ChatGPT and Claude Opus, were used during the preparation of this thesis to assist with language editing, document organization, and LaTeX preparation. All AI-assisted text and calculations were reviewed and verified by the author, who assumes full responsibility for the accuracy and content of this thesis.

Table of Contents

Page

Abstract ii

Acknowledgments iii

Table of Contents iv

List of Tables viii

List of Figures x

Chapter One: Introduction and Background 1

1.1 Motivation: Implantable Biosignal Processors 1

1.1.1 From neural recording to a local closed loop 1

1.1.2 Why a fixed on-chip classifier is insufficient 2

1.2 Challenges of On-Chip Adaptation 5

1.2.1 Stability, plasticity, and representational drift 5

1.2.2 Rare events and changing class proportions 6

1.2.3 Limited, delayed, and expensive labels 7

1.2.4 The cost of plasticity 7

1.3 Spiking Neural Networks 7

1.3.1 Poisson input encoding 8

1.3.2 Leaky integrate-and-fire dynamics 9

1.3.3 When spikes save energy 10

1.4 Spike-Timing-Dependent Plasticity 11

1.5 Baseline Network 14

1.5.1 Receptive-field learning and population competition 14

1.5.2 Three persistent plastic states 16

1.5.3 Rates, proportions, assignments, and confidence 17

1.5.4 Controlled validation with MNIST 19

1.6 Thesis Scope and Contributions 19

1.6.1 Thesis organization and validation boundaries 20

Chapter Two: Confidence-Based Selective Adaptation 24
2.1 Stability–Plasticity and Selective Learning 24
2.2 Plastic-State Ablation and Timescale Separation 27
2.3 Confidence as an Internal Supervision Signal 29
2.4 Confidence-Conditioned Ablation and the Dual-Path Rule 30
2.5 Sample-Relative Selectivity Gating 32
2.6 Labelling and Supervision 34
2.7 Results 35
2.7.1 Evaluation protocol 35
2.7.2 Robustness under covariate shift 36
2.7.3 Weight-update pressure 37
2.8 Hardware Implications and Scope of the Estimate 38
2.9 Discussion and Limitations 40
2.10 Chapter Summary 42
Chapter Three: Class Imbalance in Spiking Neural Networks 44
3.1 Prior Work on Imbalanced Learning 44
3.2 Problem Formulation and the V2 Network 46
3.2.1 Rare-event task and evaluation measures 47
3.2.2 Per-neuron selectivity and maturity 47
3.3 Diagnosis of Failure Under Severe Imbalance 48
3.4 Imbalance-Robust Plasticity Mechanisms 48
3.4.1 Drive-gated first-spike competition 49
3.4.2 State-conditioned lateral inhibition 50
3.4.3 Adaptive-threshold-modulated assignment memory 51
3.4.4 Consensus-gated weight consolidation 52
3.5 Topographic Organization and Window-Restricted Inference 52
3.6 Experimental Methodology 54
3.7 Results 55
3.8 Discussion 57

3.8.1 Implications for online adaptation . . . 58
3.8.2 Hardware implications . . . 58
3.8.3 State-aware early termination as future work . . . 58
3.9 Chapter Summary . . . 59
Chapter Four: FPGA Hardware Realization . . . 60
4.1 Overview . . . 60
4.2 Related Hardware Work . . . 60
4.3 Quantization . . . 62
4.4 Architecture . . . 63
4.4.1 System Integration . . . 64
4.4.2 Weight Memory and Banking . . . 68
4.4.3 Inference Spine . . . 69
4.4.4 Learning Engine . . . 70
4.4.5 Readout Head . . . 72
4.4.6 Control FSM. . . . 73
4.4.7 Cycle Budget . . . 73
4.5 Results . . . 74
4.5.1 Measurement Setup . . . 74
4.5.2 Functional Validation . . . 76
4.5.3 Neuron-Count Sweep . . . 76
4.5.4 Resource Utilization . . . 78
4.5.5 Power and Energy . . . 79
4.5.6 Comparison with Prior Work . . . 79
4.6 Discussion . . . 82
4.6.1 Limitations . . . 82
4.7 Chapter Summary . . . 83
Chapter Five: Conclusion and Future Work. . . . 84
5.1 Conclusion . . . 84
5.2 Statement on Contributions . . . 85

5.3 Future Work 86

Bibliography 87

List of Tables

1.1 Representative movement toward local and adaptive processing. “On-chip adaptation” does not by itself imply an unlabeled continual-learning setting. 4

1.2 Persistent plastic states of the baseline network. 16

1.3 Parameters of the original Diehl–Cook/BindsNET baseline used for the original-model experiments in this thesis. Derived per-step decay factors connect the implementation with the equations in this chapter. 22

1.4 Interpretation and limits of the controlled validation environment. 23

2.1 Persistent plastic states independently controlled in the timescale ablation. The table describes the baseline network only; confidence-conditioned update regions are derived later in Section 2.4. 27

3.1 State-conditioned inhibition gains used in the implementation 51

3.2 Failure-to-control mapping and state cost of the four mechanisms 53

3.3 Chapter-specific experimental and mechanism parameters 55

3.4 Operating points at 1:1000; detection values are simulation results and energy values are analytical estimates 57

4.1 Fixed-point widths. The 16-bit weight fraction retains headroom for graded depression; the membrane and threshold fractions are set by the once-per-sample threshold leak, and the remaining widths are bounded structurally. 64

4.2 Cycle budget per sample at $N = 400$, $T = 250$. Inference runs the per-timestep block; learning adds the trace decay each timestep and the STDP and normalization at the sample boundary. 74

4.3 On-chip neuron-count sweep, trained from scratch on-chip, against the Diehl and Cook baseline [44]. Latency, dynamic power, and energy are per sample; power and energy are the VCCINT dynamic figures averaged over a run, and the first samples of a run cost more time and energy as the network settles. The displayed accuracies are best observed values obtained with unequal training budgets, reported explicitly in the epochs column. The $N = 400$ LUT entry is from the sweep-specific placed build (147,222 LUTs, 53.7%); the final main implementation used 140,339 LUTs (51.2%), as reported separately in Section 4.5.4. The Diehl and Cook value at $N = 700$ is interpolated, since they report 100, 400, and 1,600 neurons. 77

4.4 Per-module resource utilization at $N = 400$, from the synthesized hierarchical report. The learning engine is the cost of on-chip plasticity; the inference spine runs in both modes. LUT figures include logic shared between modules and so sum slightly above the synthesized 146,023-LUT total. The separately reported final main implementation uses 140,339 LUTs, approximately 74k FFs, 803 DSPs, and 209 BRAM tiles. 78

4.5 Architecture and resource comparison for the implemented MNIST configurations. The network column gives input–hidden–output dimensions where applicable; E, I, and T denote excitatory, inhibitory, and teacher neurons. "n.r." means not reported. 80

4.6 Performance comparison with representative FPGA STDP accelerators. Accuracies are for MNIST at each reference's stated configuration; latency and energy are per image. This work's energy is measured on the board, whereas prior values are estimates or do not state a measurement method. "n.r." means not reported and "—" means not applicable. This work excludes Poisson-encoding time. 81

List of Figures

1.1 Top-level architecture of a wireless implantable closed-loop neurostimulator. The highlighted digital signal-processing block is the focus of this thesis: it converts recorded neural activity into local decisions that can drive stimulation and reduce the information sent through the telemetry link. Adapted from [10, Fig. 5]. 3

1.2 Challenges of online on-chip adaptation. The four constraints are stability–plasticity and representational forgetting, rare-event class imbalance, unavailable or delayed deployment labels, and the memory and computation cost of keeping an on-chip model plastic. 6

1.3 LIF-neuron dynamics with an adaptive threshold. Aligned traces show (a) input spikes and accumulated synaptic drive, (b) passive membrane leakage and integration, (c) threshold crossing, output spike, hard reset, and the refractory interval, and (d) the post-spike increase and slow decay of the adaptive threshold. 11

1.4 Pair-based and trace-based views of STDP. (a) The conventional timing window under the sign convention $\Delta t_{\text{spk}} = t_{\text{post}} - t_{\text{pre}}$ used in Equation (8). A presynaptic spike before a postsynaptic spike gives $\Delta t_{\text{spk}} > 0$ and potentiation; the reverse order gives depression. (b) The PostPre realization shows two spike-pair examples whose traces decay naturally between them. A postsynaptic spike samples the presynaptic trace, whereas a presynaptic spike samples the postsynaptic trace. The bottom axis shows the signed trace factors scaled by η_+ and η_- in Equation (10). 13

1.5 Original competitive STDP-SNN architecture. Poisson-encoded image pixels connect all-to-all to the excitatory population. Excitatory neurons drive paired inhibitory neurons one-to-one, and each inhibitory neuron projects to every excitatory neuron except its paired source, thereby producing lateral inhibition. Reproduced from [44, Fig. 1]. 15

1.6 Learned input-to-excitatory weights for a network with 100 excitatory neurons after 40,000 training examples. Each 784-dimensional weight vector is reshaped into a 28×28 receptive field and arranged in a 10×10 mosaic. Reproduced from [44, Fig. 2(a)]. 17

2.1 Freeze-after-epoch-1 ablation during binary one-versus-rest pretraining. Within each run, sensitivity refers to the selected event digit and specificity to the nine digits pooled into the background class. Freezing the synaptic weights W after the initial epoch has little effect, whereas freezing the adaptive thresholds θ or firing-rate tracker r degrades performance. Error bars report standard deviation over the repeated experiments. 28

2.2 Distribution of correct and incorrect predictions across confidence bins on a held-out post-pretraining test set. Labels are used only to evaluate correctness in this diagnostic. Correct decisions become more prevalent as c increases, supporting confidence as a label-free reliability signal for update routing. 30

2.3 Confidence-conditioned state ablation. Each bar is the performance change caused by freezing one state relative to the all-state configuration on that path. The confident and non-confident regimes have different state requirements, motivating the dual-path rule. 32

2.4 Control flow of the confidence-conditioned dual-path policy. The network first performs inference and forms normalized class votes. High- and low-confidence samples enable different plastic states, while the intermediate band performs inference without a state update. The high-confidence branch produces candidate weight commits; the sample-relative selectivity refinement is introduced separately in Section 2.5. 33

2.5 Minority-class sensitivity, accuracy, and specificity under rotation, translation, and zoom shifts. The confidence-gated dual-path rule provides the highest mean sensitivity in each regime. Accuracy and specificity improve under rotation and translation; under zoom, the sensitivity gain is accompanied by a small decrease in mean specificity. Error bars report standard deviation over the repeated experiments. 37

2.6 Update pressure over a 60,000-sample online run. Confidence routing removes most weight commits, while the sample-relative selectivity comparison suppresses additional redundant commits during the early active-adaptation phase. Both gated configurations settle far below the naive all-state baseline. 38

2.7 Estimated on-chip learning cost. (a) Average update activity for weight commits and non-confident θ and r updates. (b) Weight-update memory energy obtained by applying a published 45-nm access model to the measured activity. These values are analytical estimates, not FPGA or silicon power measurements of the proposed controller. 39

2.8 Preliminary neuron-participation diagnostic on the held-out post-pretraining test set. (a) Correct and incorrect proportions versus the number of neurons involved. Percentages above the bars give accuracy within each participation count. (b) Correct and incorrect proportions across four equal-width confidence intervals, normalized within the one-neuron subset. The one-neuron subset is less reliable overall than the higher-participation bins, but its high-confidence decisions are substantially more accurate, motivating joint rather than independent use of participation and confidence. 42

3.1 Diagnosis of the baseline as imbalance increases from 1:10 to 1:1000. (a) Minority-class sensitivity decreases across the imbalance sweep. (b) The number of minority-assigned neurons falls from 68 to 27, showing a loss of minority representational capacity. (c) Between 1:10 and 1:1000, the share of minority neurons in the high-selectivity, high-maturity reliable group falls from 49% to 30%. Over the same change, the low-maturity/high-selectivity group falls from 22% to 0%, the low-maturity/low-selectivity group rises from 29% to 52%, and the high-maturity/low-selectivity group rises from 0% to 18%. (d) The fraction of majority-assigned neurons with a near-duplicate receptive field rises from approximately 10% to 26%, indicating that the capacity lost by the minority is used increasingly to duplicate majority representations. 49

3.2 The four proposed mechanisms: drive-gated first-spike competition, state-conditioned lateral inhibition, adaptive-threshold-modulated assignment memory, and consensus-gated weight consolidation. 50

3.3 Minority-class sensitivity and majority-class specificity versus imbalance ratio. Bands show variation across five seeds. 55

3.4 Selectivity–maturity population at 1:1000. Baseline minority neurons are dispersed across the groups; the proposed mechanisms concentrate them more strongly in the reliable group. . 56

3.5 Minority-sensitivity degradation and normalized modeled inference energy versus topographic window size at imbalance ratios 1:10, 1:100, and 1:1000. The selected 9×9 window gives $3.7\times$ lower energy and approximately 6% relative sensitivity degradation at 1:1000. 57

4.1 Measured inference accuracy of the reference model after requantizing the trained $N = 400$ checkpoint. The translucent marker at $F = 13$ is an analytical reference: it is the lowest precision at which the rounded full-trace depression coefficient remains non-zero, not the result of a learning-precision sweep. 63

4.2 System-level integration on the ZCU102. The processing system (PS) reads samples from an SD card into DDR, Poisson-encodes each image, and streams the spike rasters to the programmable logic (PL) over AXI-DMA; the accelerator (`snn_fpga_top`) returns a per-sample decision. 66

4.3 Block diagram of the accelerator (`snn_fpga_top`): the inference spine (banked weight memory, neuron array, winner-take-all, recurrent inhibition, adaptive threshold), the learning engine (pre-/post-synaptic trace units and the STDP weight-update engine with normalization), the readout head (spike counters, rate/proportion/assignment tables, prediction and confidence), and the control FSM. Dashed outlines mark the blocks gated by the learn-enable signal. The weight-memory block is labelled per bank (1.56 kB, one neuron's 784 weights at 16 bits); the full matrix at $N = 400$ is about 627 kB. 67

4.4 The two access patterns of the banked weight memory. 69

4.5 Measurement setup: the ZCU102 board connected to a host PC over USB-JTAG (programming and debug) and USB-UART (result readout). 75

4.6 Per-rail power at $N = 400$, inference versus learning, with the static (idle) base and the dynamic component stacked. The accelerator's dynamic power sits on VCCINT; the board total is dominated by the processing-system static rails. 79

1 Introduction and Background

Long-term biomedical monitoring presents a difficult combination of requirements. Signals must be acquired continuously, clinically relevant events may occur only rarely, and the relationship between a recorded waveform and the event of interest can vary between people and over time. At the same time, an implantable device has a strict power, area, and thermal budget [1, 2]. Sending every recorded sample to an external receiver moves the computational burden outside the body, but it does not remove the energy cost of the high-rate wireless link [3]. Fixing a classifier at design time avoids on-device training, but makes the decision rule increasingly dependent on how well a finite development dataset represents the patient and the conditions encountered after deployment. These constraints motivate a processor that both makes decisions locally and changes only the internal state needed to remain useful.

This thesis studies that objective through a spiking neural network (SNN) trained by spike-timing-dependent plasticity (STDP). The central premise is that online learning need not mean keeping every model variable continuously plastic. The baseline network contains three persistent state classes—synaptic weights, adaptive neuronal thresholds, and class-conditional firing statistics—whose sizes, functions, and natural timescales are different. Treating those states separately creates opportunities to preserve representations, protect rare classes, and suppress costly memory updates. This chapter develops the biomedical motivation, introduces the SNN and STDP background required by the rest of the thesis, defines the baseline learning architecture, and states the scope of the algorithmic and hardware validation.

1.1 Motivation: Implantable Biosignal Processors

1.1.1 From neural recording to a local closed loop

A closed-loop neural interface observes physiological activity, estimates the patient's current state, and uses that estimate to trigger a response. In an epilepsy application, for example, the signal path can include neural electrodes, a recording front end, data conversion, seizure detection or prediction, and a neurostimulator [4, 5, 6, 7]. The same architectural pattern extends to other always-on biosignal applications [8]: sensing produces a continuous stream, while the output of interest is usually a much lower-rate event or control decision. Reviews of closed-loop neurostimulators describe on-chip signal processing as the element that connects high-resolution recording to timely, individualized stimulation [9, 10].

A recent review of high-density brain-implantable devices likewise identifies real-time handling and wire-

less transmission of the recorded data as system bottlenecks, motivating hardware-efficient signal reduction and processing inside the implant [11, 12]. At the acquisition boundary, the direct neural LC-ADC study in [13] examines the feasibility and circuit trade-offs of amplifier-free, input-adaptive conversion directly at neural electrodes. A complementary two-phase implementation in [14] reuses a single comparator to reduce converter area for time-sparse biosignals.

Where the loop is closed determines how energy and information move through the system. An external processor provides more computational resources and can be upgraded easily, but requires a wireless path capable of carrying either raw signals or a substantial intermediate representation. Closing more of the loop in the implant increases local computation but allows the radio to carry compact detections, status information, or occasional diagnostic segments instead. Salam et al. formulate this as a direct computation–communication trade-off in implantable neurological devices [15]. Sayedi and Kassiri identify wireless telemetry as the dominant power consumer in many representative multichannel recording systems; the examples in their review assign approximately 73–96% of total system power to the transmitter [16]. These figures are design examples rather than a universal range, but they show why increasing channel count by simply increasing the telemetry rate is difficult.

Figure 1.1, adapted from the top-level closed-loop neurostimulator architecture in [10, Fig. 5], summarizes the resulting system argument. Local processing does not make communication unnecessary: clinicians still require configuration, summaries, and selected recordings. It instead changes the default dataflow. High-rate samples remain inside the device, while a low-rate decision can drive stimulation or compact telemetry. Local decisions also avoid a round trip through an external receiver on the critical event path. The useful question is therefore not whether communication or computation should be eliminated, but how much information can be reduced safely before transmission and whether the local model can remain reliable after deployment.

1.1.2 Why a fixed on-chip classifier is insufficient

Biomedical classifiers have moved progressively closer to the sensor [17]. This trend has produced highly optimized processors for seizure detection and prediction [18, 19], including support-vector-machine (SVM) accelerators with on-chip training and model adaptation [20, 21], patient-independent processors trained from multi-patient databases [22], and source-assisted retraining intended to reduce the collection of test-patient seizure data [23]. These designs establish that useful intelligence can be placed next to a biomedical sensor. They also reveal that “adaptive,” “patient-independent,” and “zero-shot” refer to different operating

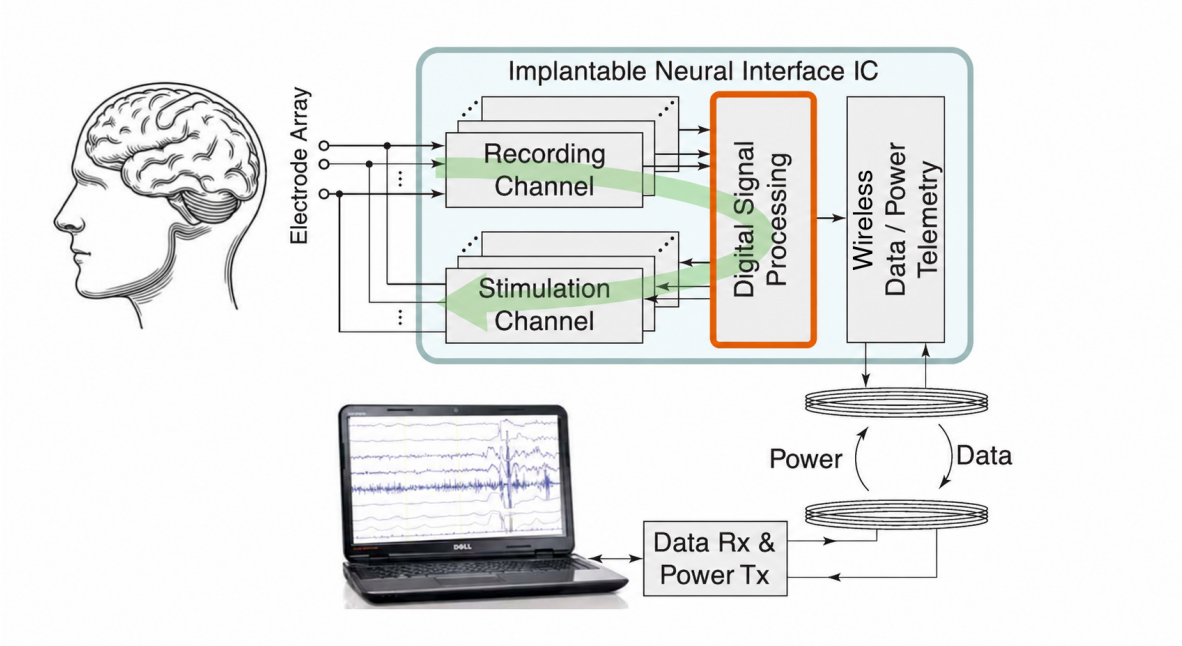


Figure 1.1: Top-level architecture of a wireless implantable closed-loop neurostimulator. The highlighted digital signal-processing block is the focus of this thesis: it converts recorded neural activity into local decisions that can drive stimulation and reduce the information sent through the telemetry link. Adapted from [10, Fig. 5].

assumptions.

The biomedical relevance of local SNN inference is demonstrated more directly by a patient-specific spiking CNN reporting 9.5-ms latency and 6.2 μJ per inference [24], and by a later fully-spiking EEG seizure detector whose encoding, quantization, neuron dynamics, and hardware realization were customized jointly [25]. These works establish low-cost SNN inference on biomedical signals, but their models are trained in software rather than adapted without labels after deployment; the latter is the distinct problem addressed in this thesis. The SciCNN processor, for example, avoids patient-specific classifier retraining by training from pre-existing labeled databases, but it performs a one-time on-chip calibration of its classifier state [22]. The zero-shot-retraining processor in [23] requires no seizure recording from the test patient, but it still uses non-seizure data from that patient together with seizure data from the source population. The SVM designs perform a supervised optimization or use labels generated by a preceding model during adaptation [20, 21]. These are valuable and practical points in the design space; they should not, however, be interpreted as long-term label-free continual learning.

Neuromorphic processors provide a complementary direction. RAINE demonstrates a fabricated multi-topology SNN processor with on-chip trace-based rewarded STDP, and ReckOn demonstrates end-to-end learning of temporal tasks using a local approximation to backpropagation through time [26, 27]. Their

Table 1.1: Representative movement toward local and adaptive processing. "On-chip adaptation" does not by itself imply an unlabeled continual-learning setting.

Work	Processing model	Local learning or calibration	Supervision assumption relevant here
Active-learning SVM [20]	SVM seizure detector	Active sample selection and on-chip SVM training	Labeled SVM objective; demonstrates efficient local model adaptation
Adaptable neural processor [21]	SVM seizure predictor	On-chip ADMM-based adaptation	Supervised objective, with model-generated labels available for online adaptation
SciCNN [22]	Patient-independent CNN processor	Short on-chip classifier calibration	Labeled multi-patient pretraining; zero patient-specific retraining is not zero pretraining
Zero-shot-retraining processor [23]	Adaptive seizure detector	On-chip source-assisted retraining and channel selection	No test-patient seizure data, but target non-seizure and source seizure data are used
RAINE [26]	Multi-topology SNN processor	Trace-based rewarded STDP	A reward/label supervises the weight update
ReckOn [27]	Recurrent SNN processor	Approximate e-prop over long temporal dependencies	Error learning signals supervise end-to-end learning

measured implementations show that learning-capable spiking hardware can operate under edge budgets. RAINE's reward indicates whether a prediction agrees with a label, and ReckOn's approximate e-prop rule uses error learning signals; neither is an example of unlabeled pair-based STDP adaptation. Table 1.1 organizes these representative processors by the assumption most relevant to this thesis rather than by headline accuracy, since their datasets, technologies, and tasks are not directly comparable.

The remaining difficulty is deployment variability. Neural and other physiological signals differ across patients [28], electrode locations, recording sessions, activity states, and acquisition conditions [29, 30]. A patient-specific model can compensate for some of this variability [31], but collecting enough labeled pathological events can require a lengthy monitoring or hospitalization period precisely because those events are rare [22, 23]. After deployment, labels may be delayed, sparse, or absent: an implant observes a signal immediately, whereas a reliable annotation can require clinical review and context unavailable to the device. Repeating a full supervised calibration whenever the distribution changes is therefore inconsistent with autonomous long-term operation.

Continual learning addresses the more general problem of updating a model as data arrive sequentially rather than assuming a stationary, fully labeled training set. It must balance plasticity, the ability to acquire information from the current stream, against stability, the retention of useful prior knowledge [32, 33]. The setting studied here keeps the label space fixed while the input distribution and class frequency can change;

it is closest to online domain adaptation rather than task-incremental or class-incremental learning. No task identifier announces a change. This distinction matters because the device cannot select a separately stored model whenever a new patient or session begins.

Adaptation should occur locally for the same reason that inference should occur locally: routinely exporting the stream reintroduces the telemetry cost, while requiring an external learning service breaks autonomous operation. Local adaptation also permits the device to respond at the rate at which its environment changes. However, on-device learning is more demanding than inference. It must retain intermediate state, decide which observations are trustworthy, and write persistent model variables. Thus a low-complexity classifier does not automatically yield a low-cost adaptive processor.

SNNs are promising in this setting because information is represented by discrete events and learning can be expressed using state local to a synapse or neuron. Sparse spike-driven arithmetic can reduce unnecessary operations on appropriate hardware, while STDP avoids a global gradient computation and the storage of an entire temporal computation graph [34]. The advantage is conditional: a dense spike train, an inefficient time-stepped implementation, or uncontrolled synaptic rewriting can erase it. Moreover, local plasticity does not by itself solve continual learning. Prior SNN methods address forgetting through adaptive plasticity, controlled forgetting, and gradual unsupervised adaptation [35, 36, 37, 38], but the stability–plasticity problem remains. This thesis therefore asks not merely whether an SNN can learn locally, but which of its states should learn from each observation and at what hardware cost.

1.2 Challenges of On-Chip Adaptation

Moving learning from a workstation to a continuously operating device changes both the information available to the algorithm and the resources available to execute it. Four recurring challenges are shown in Figure 1.2: the input distribution can change, clinically important classes can be rare, deployment labels can be unavailable, and every model update consumes storage, computation, and energy. This section states those challenges at the application and continual-learning level. Their specific realization in the SNN is deferred until after the neuron model, STDP rule, and baseline architecture have been introduced.

1.2.1 Stability, plasticity, and representational drift

If the model changes too little, it remains specialized to its pre-deployment distribution and cannot follow a new patient or session. If it changes too readily, new observations can disturb information that remains useful.

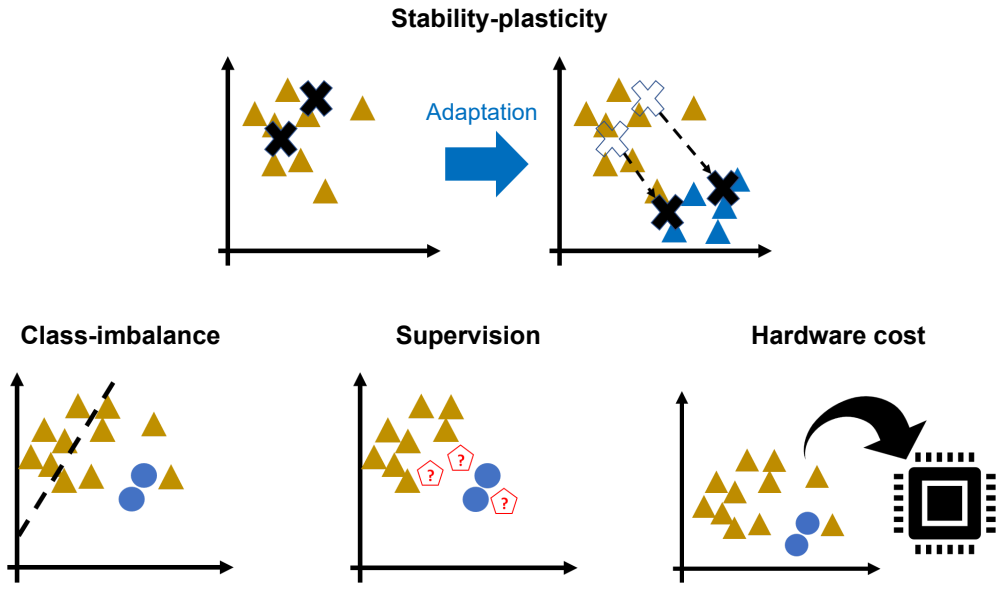


Figure 1.2: Challenges of online on-chip adaptation. The four constraints are stability–plasticity and representational forgetting, rare-event class imbalance, unavailable or delayed deployment labels, and the memory and computation cost of keeping an on-chip model plastic.

This tension is the stability–plasticity dilemma in continual learning [32]. Its most severe manifestation is catastrophic forgetting, in which learning a new task causes a large loss on previously learned tasks [33]. Patient and session changes are not necessarily new tasks: the label space may remain fixed while the input distribution drifts. A deployment method must therefore remain responsive to non-stationarity without assuming that every change announces a new task or justifies replacing the existing model. The experiments in this thesis study this narrower setting of a continuing covariate shift and representational drift; they do not claim a general solution to catastrophic forgetting over arbitrary task sequences.

1.2.2 Rare events and changing class proportions

In continuous monitoring, background observations can outnumber the event of interest by orders of magnitude. Overall accuracy can then be misleading: a detector may classify the abundant background correctly while missing a clinically unacceptable fraction of rare events. The seizure-processing review in [10] makes the same point for evaluation metrics, noting that a large number of true negatives can keep specificity high even when false alarms are substantial. More generally, class imbalance changes what a learner observes and how its performance should be measured [39]. In a stream, the class ratio may itself vary with patient state and time, so a method cannot always rely on a fixed resampling schedule or a known prior ratio [40]. Sensi-

tivity, false-alarm behavior, and retention of minority information must consequently be examined separately from aggregate accuracy.

1.2.3 Limited, delayed, and expensive labels

Continual supervised learning assumes that a trustworthy target accompanies each new observation. That assumption is difficult to satisfy in long-term biomedical monitoring. The signal is available immediately, but an annotation can require an expert review, additional clinical context, or observation of an event that occurs infrequently. In seizure monitoring, collecting patient-specific seizure recordings for training can require costly and time-consuming hospitalization [22, 23]. Labels obtained after deployment may therefore be sparse or delayed rather than synchronized with the stream. Unsupervised and self-supervised continual-learning methods are motivated by this gap [32]. Internally generated targets can nevertheless introduce their own errors. A practical device should distinguish initial supervised calibration, optional later clinical feedback, and the label-free operation that must occur between them.

1.2.4 The cost of plasticity

Training introduces costs that are absent when a frozen model performs inference. Intermediate learning state must be retained, update rules must be evaluated, and modified parameters must be written. Continual-learning methods based on replay or large auxiliary models can add memory and computation beyond the inference network [32]. Data movement can also cost substantially more energy than elementary arithmetic, particularly when training repeatedly reads and rewrites a large parameter array [41]. Hence an algorithm suitable for a workstation may remain unsuitable for an implant even if its inference path is efficient. The relevant measures include learning-state capacity, update frequency, memory traffic, latency, and energy, not classification performance alone.

These four challenges motivate the event-driven and local learning background in the next sections. Once the common network has been defined, Section 1.6 states which challenge is evaluated by each thesis chapter and which interactions are left for future validation.

1.3 Spiking Neural Networks

An SNN represents communication between neurons as discrete spikes while retaining a time-dependent state within each neuron. This distinguishes it from a conventional feedforward artificial neural network,

whose activations are normally evaluated once per layer for a static input. Spiking networks are consequently often described as a third generation of neural-network models [42]; their neuron dynamics, codes, and learning rules are reviewed in [43, 34]. The state of a spiking neuron may evolve continuously or at discrete simulation steps, but information is communicated to connected neurons only when an output spike is generated.

The timing of spikes can itself carry information, as can their order, latency, or count over an observation interval [43]. This thesis uses a rate-coded, time-stepped SNN. Input magnitude controls the expected number of input spikes, the relative timing of pre- and postsynaptic events drives STDP, and the classification readout uses the number and class composition of output spikes. Thus the learning rule remains timing dependent even though the final class decision is rate based.

1.3.1 Poisson input encoding

A non-spiking sample must first be converted to a spike train. Let M be the number of input channels, $u_i \in [0,1]$ the normalized value of channel $i \in \{1,\dots,M\}$, γ the maximum encoded firing rate in spikes per second, Δt_s the duration of one simulation step in seconds, and n the integer step index. The nominal firing rate assigned to channel i is

$$\lambda_i = \gamma u_i. \tag{1}$$

The experiments generate binary input events $x_i[n] \in \{0,1\}$ with the BindsNET `PoissonEncoder`. For $\lambda_i > 0$, the encoder samples nonzero integer inter-spike intervals whose mean scales as $(\lambda_i \Delta t_s)^{-1}$ simulation steps and cumulatively converts those intervals into event times. This is the library's discrete inter-spike-interval procedure, not an independent Bernoulli trial applied at every timestep. A larger u_i therefore produces shorter expected intervals and more events, while individual spike times remain stochastic. This rate encoding is used by the adopted competitive STDP network [44, 45].

The controlled experiments use the Modified National Institute of Standards and Technology (MNIST) handwritten-digit dataset [46]. MNIST contains 60,000 training images and 10,000 test images of the digits 0–9; each image is a 28×28 grayscale array. Consequently, $M = 784$ and u_i is a normalized pixel intensity. MNIST is introduced here as the source of the encoded input, while Section 1.5 explains how the thesis converts it into controlled rare-event and distribution-shift experiments. For a future biosignal system, u_i could instead represent a waveform sample, band energy, channel feature, or event produced by an analog-

to-spike encoder [13, 47]. The subsequent learning mechanisms operate on the spike stream rather than on the semantic meaning of a pixel.

1.3.2 Leaky integrate-and-fire dynamics

The leaky integrate-and-fire (LIF) model retains the principal operations needed here: integration of synaptic input, passive leakage toward a resting voltage, threshold crossing, reset, and a refractory interval [43]. In continuous time, the input spike train of channel i can be written

$$S_i(t) = \sum_f \delta\left(t - t_i^{(f)}\right), \tag{2}$$

where t is continuous time, $t_i^{(f)}$ is the time of the fth spike from input i, and $\delta(\cdot)$ is the Dirac delta distribution. The idealized impulses in $S_i(t)$ drive a synaptic current. Between spikes, the membrane voltage $v_j(t)$ of neuron j follows the standard subthreshold relation

$$\tau_m \frac{dv_j(t)}{dt} = -\left(v_j(t) - v_{\text{rest}}\right) + R_m I_j^{\text{syn}}(t), \tag{3}$$

where τ_m is the membrane time constant, v_{rest} is the resting voltage, R_m is the membrane resistance, and $I_j^{\text{syn}}(t)$ is the total synaptic current applied to neuron j.

The software and hardware models used in this thesis operate at discrete steps. At each step, the membrane voltage is first updated according to

$$v_j[n+1] = v_{\text{rest}} + \beta\left(v_j[n] - v_{\text{rest}}\right) + \mathbb{1}[h_j[n] = 0] \sum_{i=1}^{M} w_{ij} x_i[n], \tag{4}$$

where $v_j[n]$ is the stored membrane voltage at step n, $\beta = \exp(-\Delta t_s/\tau_m)$ is the discrete leak factor, and w_{ij} is the synaptic weight from input i to neuron j. The summation is the input accumulated during that step, and the indicator suppresses this input while the neuron is refractory; membrane leakage remains active. Network-level lateral inhibition is introduced separately in Section 1.5 and enters the same update as an additional negative input.

Let $h_j[n]$ be a nonnegative refractory counter. The binary output event is

$$z_j[n+1] = \mathbb{1}[h_j[n] = 0]\,\mathbb{1}[v_j[n+1] \geq v_{\text{thr}} + \theta_j[n]], \tag{5}$$

where v_{thr} is the fixed firing threshold and $\theta_j[n]$ is a neuron-specific adaptive threshold increment. The symbol $\mathbb{1}[A]$ is an indicator: it equals one when statement A is true and zero otherwise. It is not a Dirac delta. Equation (5) is the binary representation used by a discrete simulator or digital circuit, whereas the corresponding continuous-time output spike train is $S_j(t) = \sum_f \delta(t - t_j^{(f)})$.

When $z_j[n+1] = 1$, the membrane voltage is set to the reset voltage v_{reset}, and the refractory counter is initialized to H_{ref} steps. Otherwise, the voltage obtained from Equation (4) is retained and the counter is reduced toward zero. During the refractory interval, incoming synaptic events are discarded, but the membrane continues to leak from v_{reset} toward v_{rest}.

The adaptive component θ_j is a slow, neuron-specific addition to the fixed threshold. Its continuous-time dynamics can be expressed using the output Dirac train,

$$\frac{d\theta_j(t)}{dt} = -\frac{\theta_j(t)}{\tau_\theta} + \theta_+ \sum_f \delta\left(t - t_j^{(f)}\right), \tag{6}$$

where τ_θ is the adaptation time constant and θ_+ is the increment produced by an output spike. The discrete equivalent is

$$\theta_j[n+1] = \rho_\theta \theta_j[n] + \theta_+ z_j[n+1], \qquad \rho_\theta = \exp(-\Delta t_s/\tau_\theta). \tag{7}$$

A neuron that fires repeatedly therefore raises its future effective threshold, $v_{\text{thr}} + \theta_j$, whereas a quiet neuron's increment decays. This homeostatic mechanism, used in the adopted competitive network [44], limits persistently high firing and encourages different neurons to become active for different inputs.

1.3.3 When spikes save energy

The absence of an input spike creates an opportunity to skip a synaptic accumulation, but the opportunity must be realized by the architecture. A digital implementation must still store and update membrane state, evaluate several timesteps, and move weights and state through its memory hierarchy. Ottati et al. show that, for current digital accelerators, these costs can outweigh multiply-free spike processing on static-image tasks; the surveyed ANN accelerators outperform their SNN counterparts in both efficiency and accuracy in that regime [48]. Their analysis finds a more competitive case for temporal workloads and identifies event-based sensors and hybrid ANN–SNN systems as important directions.

This qualification is particularly relevant because MNIST is a static image dataset. In this thesis it provides a controlled way to study learning state, imbalance, and distribution shift; it does not by itself demonstrate

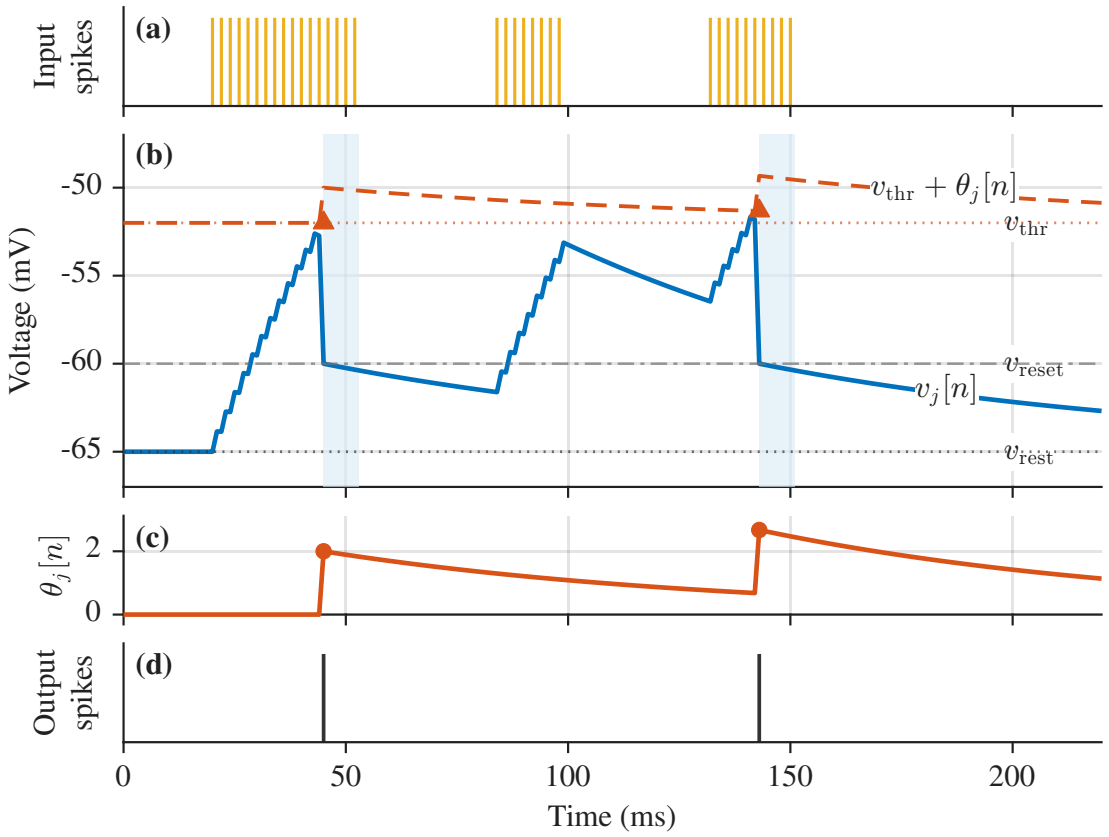


Figure 1.3: LIF-neuron dynamics with an adaptive threshold. Aligned traces show (a) input spikes and accumulated synaptic drive, (b) passive membrane leakage and integration, (c) threshold crossing, output spike, hard reset, and the refractory interval, and (d) the post-spike increase and slow decay of the adaptive threshold.

that an SNN is more efficient than an ANN. Fabricated learning processors nevertheless show that sparse, local spiking computation can operate within edge budgets when the workload, memory hierarchy, and learning datapath are designed together [26, 27]. Efficiency is therefore treated as a measured property of an implementation, not an intrinsic property of the word "spiking."

1.4 Spike-Timing-Dependent Plasticity

STDP is a family of local learning rules in which synaptic change depends on the relative timing of presynaptic and postsynaptic spikes. Experimental observations and phenomenological STDP models are reviewed in [49]; connections to modern SNN training are discussed in [34]. Let t_{pre} and t_{post} denote a presynaptic and postsynaptic spike time, respectively, and define $\Delta t_{\text{spk}} = t_{\text{post}} - t_{\text{pre}}$. A conventional pair-based learning

window is

$$\Delta w(\Delta t_{\text{spk}}) = \begin{cases} A_+ \exp(-\Delta t_{\text{spk}}/\tau_+), & \Delta t_{\text{spk}} > 0, \\ -A_- \exp(\Delta t_{\text{spk}}/\tau_-), & \Delta t_{\text{spk}} < 0, \end{cases} \tag{8}$$

where $A_+, A_- > 0$ set the maximum potentiation and depression magnitudes, and $\tau_+, \tau_- > 0$ set the two timing-window widths. Under this sign convention, a presynaptic spike before a postsynaptic spike gives $\Delta t_{\text{spk}} > 0$ and potentiates the synapse; the reverse order gives $\Delta t_{\text{spk}} < 0$ and depresses it. The precise shape and even sign of measured timing windows can depend on the synapse and experimental conditions, so Equation (8) is a standard phenomenological model rather than a universal biological law [49].

The implementation need not retain every spike time. Let $x_i^{\text{pre}}[n] \in [0,1]$ be the trace associated with presynaptic input i, and $x_j^{\text{post}}[n] \in [0,1]$ the trace associated with postsynaptic neuron j. The binary events are $x_i[n]$ and $z_j[n]$, as defined in Section 1.3. A trace is set to one by its event and otherwise decays:

$$\begin{aligned} x_i^{\text{pre}}[n+1] &= \begin{cases} 1, & x_i[n] = 1, \\ \lambda_{\text{pre}} x_i^{\text{pre}}[n], & x_i[n] = 0, \end{cases} \\ x_j^{\text{post}}[n+1] &= \begin{cases} 1, & z_j[n] = 1, \\ \lambda_{\text{post}} x_j^{\text{post}}[n], & z_j[n] = 0, \end{cases} \end{aligned} \tag{9}$$

where $0 < \lambda_{\text{pre}}, \lambda_{\text{post}} < 1$ are the per-step decay factors; for exponential traces they equal $\exp(-\Delta t_s/\tau_{\text{pre}})$ and $\exp(-\Delta t_s/\tau_{\text{post}})$, respectively, where τ_{pre} and τ_{post} are the trace time constants.

For a synapse from presynaptic input i to postsynaptic neuron j, the BindsNET implementation used by the baseline applies the additive trace-based PostPre rule

$$\Delta w_{ij}[n] = \eta_+ x_i^{\text{pre}}[n] z_j[n] - \eta_- x_i[n] x_j^{\text{post}}[n], \tag{10}$$

where w_{ij} is the synaptic weight and $\eta_+, \eta_- > 0$ are the potentiation and depression learning rates. The first term is nonzero when neuron j spikes and samples recent presynaptic activity. The second is nonzero when input i spikes and samples recent postsynaptic activity. After applying the change, the bounded weight is

$$w_{ij}[n+1] = \text{clip}(w_{ij}[n] + \Delta w_{ij}[n], w_{\min}, w_{\max}), \tag{11}$$

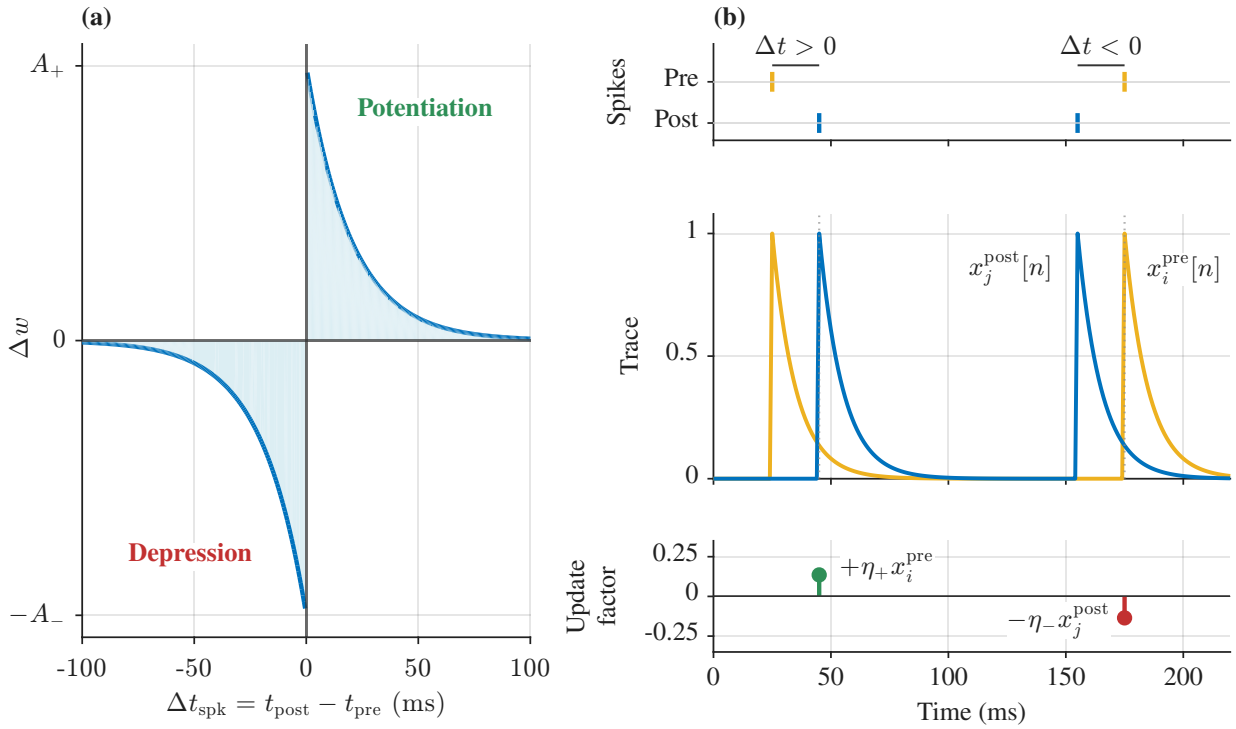


Figure 1.4: Pair-based and trace-based views of STDP. (a) The conventional timing window under the sign convention $\Delta t_{\text{spk}} = t_{\text{post}} - t_{\text{pre}}$ used in Equation (8). A presynaptic spike before a postsynaptic spike gives $\Delta t_{\text{spk}} > 0$ and potentiation; the reverse order gives depression. (b) The PostPre realization shows two spike-pair examples whose traces decay naturally between them. A postsynaptic spike samples the presynaptic trace, whereas a presynaptic spike samples the postsynaptic trace. The bottom axis shows the signed trace factors scaled by η_+ and η_- in Equation (10).

where $w_{\min}$ and $w_{\max}$ are the permitted lower and upper weight limits, and $\text{clip}(a,l,u) = \min(u, \max(l,a))$. Thus all variables required by an update are local to the two endpoints and their connecting synapse. The additive PostPre rule is the concrete BindsNET update used in this thesis; the principal results in [44] instead used a power-law weight-dependent rule. Neither update uses the sample's class label [45].

STDP is attractive for online operation because Equation (10) uses only local spikes and traces. For M presynaptic and N postsynaptic neurons, its trace state scales as $O(M+N)$ when traces are stored per neuron, rather than with the full history required by backpropagation through time [34]. This locality does not make continual learning safe automatically. If every eligible event remains plastic indefinitely, new observations can overwrite synaptic associations that are still useful. Several works therefore augment local SNN learning with adaptive plasticity, stop-learning, consolidation, replay, or controlled forgetting [35, 36, 37, 50]. The approach in this thesis retains the ordinary STDP rule and instead controls when its synaptic updates are eligible to occur.

1.5 Baseline Network

The baseline considered in this section is the competitive network introduced by Diehl and Cook for unsupervised digit recognition [44], implemented through the BindsNET simulation framework [45]. It contains an input population, an excitatory population, and a corresponding inhibitory population. Each excitatory neuron drives one paired inhibitory neuron through a fixed positive connection. That inhibitory neuron then suppresses every excitatory neuron except the one to which it is paired. This disynaptic lateral-inhibition path produces winner-take-all (WTA) competition while leaving the input-to-excitatory synapses plastic.

For an input dimension M and N excitatory neurons, the dense feedforward matrix is $W \in \mathbb{R}^{M\times N}$. Poisson spikes from all input channels drive the adaptive excitatory LIF neurons. A selected excitatory spike activates its paired inhibitory neuron, whose fixed outgoing connections suppress the competing excitatory neurons. Under the one-spike option, at most one excitatory neuron is selected at each timestep; if several cross threshold together, the BindsNET model draws one winner from a multinomial distribution over the candidates. In the BindsNET update order, all crossing candidates first undergo voltage reset, refractory-counter initialization, and adaptive-threshold increment; the multinomial selection then retains one communicated output spike. The one-spike rule therefore constrains emitted population activity without undoing those candidate-neuron state updates.

Figure 1.5 reproduces the original network architecture reported in [44, Fig. 1].

1.5.1 Receptive-field learning and population competition

When an excitatory neuron wins, recent input activity potentiates the corresponding connections. Inputs inconsistent with its output history are depressed, and normalization redistributes a fixed total weight. Let $\mathbf{w}_{:j} = [w_{1j},\ldots,w_{Mj}]^{\mathsf{T}}$ denote the complete incoming weight vector of excitatory neuron j. After an update and clipping, the baseline applies

$$\mathbf{w}_{:j} \leftarrow \kappa \frac{\mathbf{w}_{:j}}{\sum_{i=1}^{M} w_{ij}}, \tag{12}$$

where $\kappa > 0$ is the target L_1 sum. This prevents a neuron from winning merely because it has accumulated a larger total input weight and turns learning into a competition over how a finite synaptic budget is distributed. Divisive normalization was analyzed by Goodhill and Barrow as a constraint that causes prototypes in a competitive-learning system to spread across typical inputs [51]. In the adopted SNN, it is used to promote more equal use of the neurons [44]. Repeated wins on related samples can therefore redirect the fixed-sum

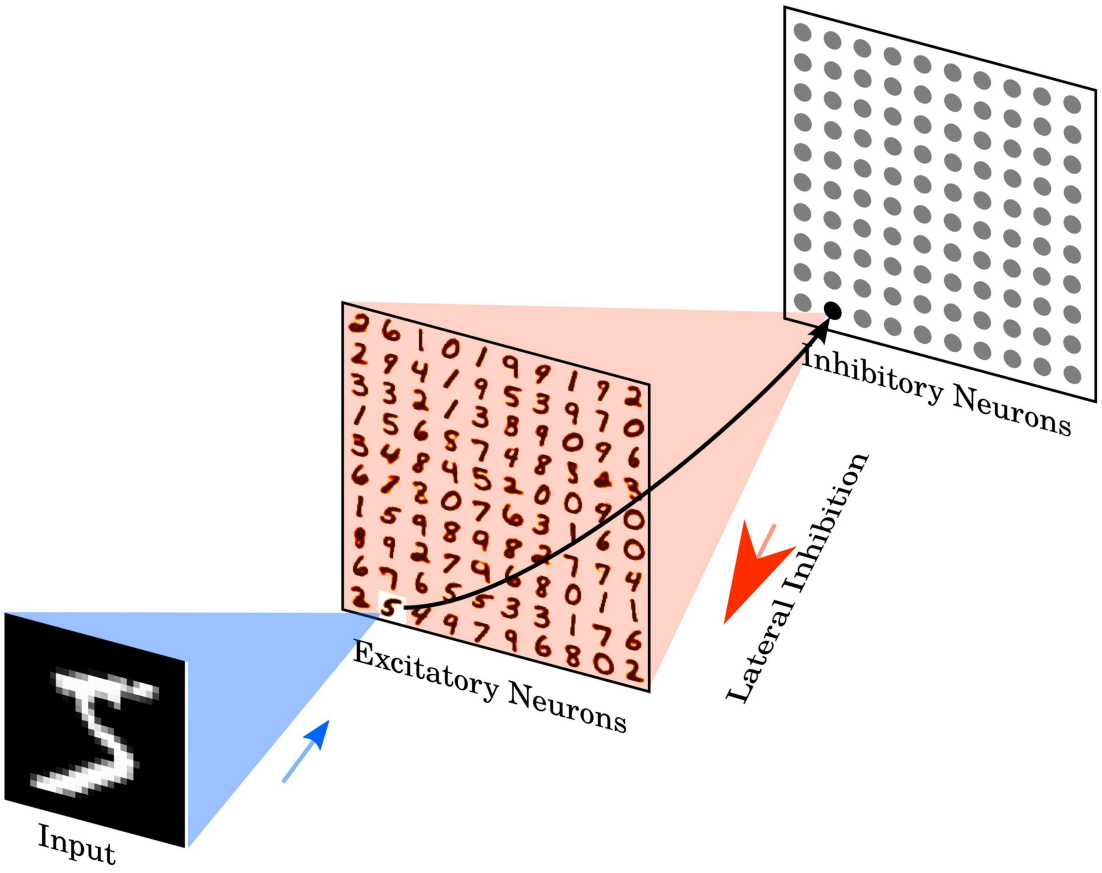


Figure 1.5: Original competitive STDP-SNN architecture. Poisson-encoded image pixels connect all-to-all to the excitatory population. Excitatory neurons drive paired inhibitory neurons one-to-one, and each inhibitory neuron projects to every excitatory neuron except its paired source, thereby producing lateral inhibition. Reproduced from [44, Fig. 1].

weight vector toward a recurring input pattern rather than simply increasing its overall magnitude. In the MNIST visualization, reshaping that vector to 28×28 pixels reveals a digit-like receptive field. The same concept is not inherently visual: with a biosignal feature vector, a receptive field would show the temporal, spectral, or channel pattern that drives the neuron.

This benefit introduces an important locality and implementation trade-off. The STDP increment in Equation (10) is event-local to one synapse and its two endpoint neurons. Exact enforcement of Equation (12) is only local at the level of the postsynaptic neuron: changing even one w_{ij} changes the normalizing denominator, so all M incoming weights of neuron j may have to be read, rescaled, and written. Normalization consequently constrains aggregate weight growth and removes total afferent strength as an accidental competitive advantage, but it also adds state access beyond the synapses directly eligible for STDP. The baseline retains this operation because it is part of the competitive learning mechanism; its additional learning traffic

Table 1.2: Persistent plastic states of the baseline network.

State	Size	Baseline update	Functional interpretation
Synaptic weights W	$O(MN)$	Trace-based STDP followed by clamping and normalization	Slow receptive-field representation; largest storage and dominant commit traffic
Adaptive thresholds θ	$O(N)$	Increment on an output spike and slow decay	Homeostatic state regulating firing and specialization
Class rates r	$O(NC)$	Accumulation or exponential retention of class-conditional spike counts	Semantic memory from which proportions, neuron assignments, selectivity, prediction, and confidence are derived

Notation: M is the number of presynaptic input channels, N is the number of excitatory neurons, and C is the number of semantic classes.

is treated explicitly as an implementation cost rather than being hidden by the term "local learning."

The original study also differs from conventional supervised ANN training in how the end of learning is specified. It does not define a validation-error early-stopping criterion of the type used to stop gradient training before overfitting [52]. Instead, a predetermined number of examples is presented, after which the learning rate is set to zero and the adaptive thresholds are fixed [44]. Its receptive-field mosaics provide a qualitative check that neurons have learned recognizable prototypes, while the reported accuracy as a function of the number of presentations provides a separate quantitative convergence check. Receptive-field inspection is therefore useful evidence about what has been learned, but it should not be described as a formal stopping rule.

Lateral inhibition couples the receptive fields. A strong winner prevents every neuron from learning the same sample, while adaptive thresholds prevent the same winner from dominating every presentation. Together they distribute representational capacity over the neuron population. That distribution is precisely what severe class imbalance can corrupt: majority samples receive more competitions and can recruit neurons that had begun to represent the minority. The mosaic in Figure 1.6, reproduced from [44, Fig. 2(a)], makes this internal representation visible by reshaping each learned weight vector into the original input geometry.

1.5.2 Three persistent plastic states

The network exposes three state classes that remain relevant after a sample ends. They are summarized in Table 1.2.

The size distinction is central. For a dense input layer, W grows with both the number of inputs and output neurons. The other two states are neuron-level. They also play different roles: W answers what input

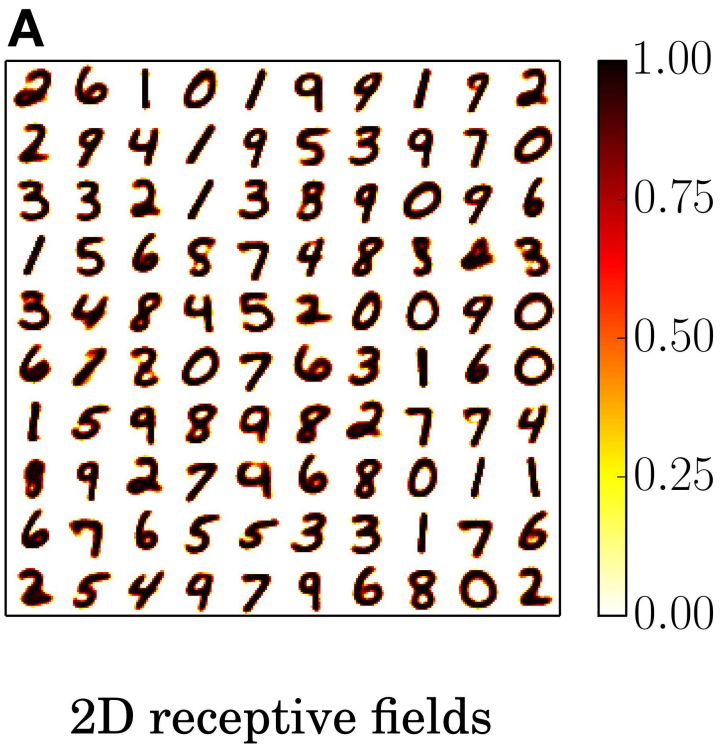


Figure 1.6: Learned input-to-excitatory weights for a network with 100 excitatory neurons after 40,000 training examples. Each 784-dimensional weight vector is reshaped into a 28×28 receptive field and arranged in a 10×10 mosaic. Reproduced from [44, Fig. 2(a)].

pattern a neuron recognizes, θ controls how readily it participates, and r records which semantic class has historically produced its activity. A policy that updates all three whenever learning is enabled assumes that these functions share one timescale and one reliability requirement. Chapter 2 tests that assumption directly.

1.5.3 Rates, proportions, assignments, and confidence

STDP discovers receptive fields without using class labels, but a classifier must map the discovered neurons to semantic classes. Let $s_j(x)$ be the number of spikes from neuron j during sample x. At a fixed label interval, the baseline accumulates a class-conditional response $r_{j,k}$ for each neuron j and class k. A convenient discrete update is

$$r_{j,k} \leftarrow \alpha r_{j,k} + \widetilde{r}_{j,k}, \tag{13}$$

where $\widetilde{r}_{j,k}$ is the newly measured class-conditional response for class k—for example, its mean spike count over the newest labeled interval—and α controls retention of prior evidence. During labeled pretraining, attribution uses the true class. During the unlabeled deployment policy of Chapter 2, the predicted class acts

as a pseudo-label when an assignment update is permitted.

The normalized class proportions and neuron assignment are

$$p_{j,k} = \frac{r_{j,k}}{\sum_{k'=1}^{C} r_{j,k'}}, \qquad a_j = \arg\max_k p_{j,k}. \tag{14}$$

A high p_{j,a_j} means that neuron j has responded predominantly to its assigned class. The full proportion vector also supports a weighted population vote. For a sample x, define

$$q_k(x) = \sum_{j=1}^{N} s_j(x) p_{j,k}. \tag{15}$$

The predicted class and confidence are

$$\widehat{y}(x) = \arg\max_k q_k(x), \qquad c(x) = \frac{q_{\widehat{y}(x)}(x)}{\sum_{k=1}^{C} q_k(x)}. \tag{16}$$

Because each proportion vector sums to one, the confidence is the fraction of the weighted population vote supporting the winning class. In the binary rare-event task, a nonzero response gives $c \in [0.5, 1]$: a value near 0.5 indicates a split vote, and a value near one indicates agreement among the active population. This definition separates decisiveness from total firing strength; multiplying all spike counts by the same factor does not change c.

Operationally, new spike counts update the class rates r; normalization produces the proportions p and assignments a; the active neurons' proportion vectors form the class scores q; and the winning share of those scores gives the confidence c. This readout procedure is important for interpreting "unsupervised" in the rest of the thesis. The feedforward representation is learned without a class label, but labels are used initially to attach semantic meaning to the output neurons. The principal online-adaptation experiments assume that those deployment labels disappear after pretraining. Their internal prediction can update selected readout state, but it is not equivalent to receiving new clinical ground truth. Any claim of label-free operation in this thesis refers to that deployment phase, not to a model that has never seen labeled data.

Table 1.3 consolidates the symbols and baseline settings introduced across Sections 1.3–1.5. Here, "baseline" means the original excitatory–inhibitory configuration instantiated for the experiments based on this network, not every constructor default of the BindsNET library [45]. This distinction matters because the constructor defaults to 100 neurons in each excitatory and inhibitory population and an inhibitory magnitude of 17.5,

whereas the thesis configuration uses $N = 400$ excitatory neurons, $N_I = 400$ inhibitory neurons, and an inhibitory magnitude of 120. Unless a study explicitly states a different value, the settings in the table apply to this original model. Parameters belonging to later architectural variants or proposed controllers are reported in their respective chapters.

1.5.4 Controlled validation with MNIST

The intended application is an implantable rare-event biosignal processor. This thesis does not train on clinical recordings, fabricate an implantable ASIC, or claim clinical detection performance. Instead, it separates mechanism validation from application validation. Poisson-encoded MNIST supplies a reproducible stream whose class frequency and covariate shift can be controlled independently. A one-versus-rest task makes one digit the rare event and combines the remaining digits into a heterogeneous background. Rotation, translation, and zoom change the input distribution without changing the class definition. These transformations do not model a particular physiological process; they are controlled perturbations used to test whether the learning mechanisms respond to distribution change.

The abstraction is reasonable for the questions asked because the proposed controls observe spikes, thresholds, weights, rates, and population decisions rather than image geometry. It also makes failure visible through receptive fields and permits extreme imbalance to be generated without waiting for rare clinical events. Nevertheless, the MNIST and FPGA results provide evidence of mechanism validity and implementation feasibility. A clinical recording study and an implantable mixed-signal ASIC remain necessary before the system-level motivation of Figure 1.1 can be claimed as an implemented medical device.

1.6 Thesis Scope and Contributions

The objective of this thesis is to determine how the plastic state of an STDP-SNN can be controlled under three constraints relevant to implantable processing. The constraints are evaluated in separate studies rather than presented as one already integrated deployment system: Chapter 2 studies unlabeled post-pretraining covariate shift, Chapter 3 studies severe class imbalance during training from scratch, and Chapter 4 measures an FPGA realization of the V2 learning baseline. The work contributes at three connected levels.

1) **Confidence-conditioned selective adaptation.** Chapter 2 decomposes the baseline into W, θ, and r, then uses freeze ablations to establish that they should not share a single plasticity schedule. It derives a

dual-path controller from the network's own confidence: confident samples consolidate selected synaptic state, non-confident samples update the smaller homeostatic and assignment state, and an intermediate band abstains. A sample-relative neuron-selectivity gate further suppresses commits to already specialized receptive fields while allowing plasticity to return when selectivity falls. Across controlled rare-event covariate shifts, the combined policy improves mean minority sensitivity relative to naive all-state adaptation and reduces weight-update pressure by approximately 80% during active adaptation and 92% at steady state. These percentages describe memory transactions in the algorithmic evaluation, not measured ASIC energy.

2) **Local protection against severe class imbalance during training.** Chapter 3 characterizes imbalance during training from scratch in the WTA-STDP network as a population-allocation failure: rare-class specialists are reassigned and the majority acquires redundant representational capacity. It introduces four local, ratio-agnostic controls acting on first-spike competition, state-conditioned inhibition, adaptive-threshold-modulated assignment retention, and consensus-gated weight consolidation. A topographic organization then restricts inference to a neighborhood anchored by the first spike. In the controlled 1:1000 condition, the plasticity mechanisms increase minority sensitivity from 41% to 72%; a 9×9 inference window retains 68% sensitivity while reducing modeled inference energy from 6.7 to $1.8\,\mu$J. The energy figures are analytical estimates rather than FPGA measurements.

3) **Measured FPGA realization of the V2 learning baseline.** Chapter 4 maps adaptive LIF dynamics, WTA competition, trace-based STDP, normalization, and the firing-rate readout to fixed-point programmable logic. The accelerator accepts Poisson-encoded inputs generated by the processing system. At the $N = 400$ design point, the implementation achieves 86.2% MNIST accuracy, $131\,\mu$s inference latency, and $292\,\mu$s learning latency. Measured programmable-logic dynamic energy is $22\,\mu$J per inference and $66\,\mu$J per learning sample. The implementation makes the learning overhead concrete and shows that weight update and normalization dominate the extra learning time.

1.6.1 Thesis organization and validation boundaries

Chapter 2 first considers what should remain plastic after the baseline has been pretrained. It develops confidence-conditioned update routing for an unlabeled covariate-shift stream and quantifies both predictive behavior and synaptic-memory traffic. Chapter 3 then asks a different question: how severe class imbalance

affects population organization while the network is being trained from scratch. Its mechanisms are not presented as a jointly validated extension of the Chapter 2 controller; their possible interaction during long-term imbalanced adaptation is discussed as future validation. The adaptation study uses the original architecture defined in Section 1.5, including its explicit inhibitory population. Chapter 3 instead uses the BindsNET V2 variant, which replaces that population with direct recurrent inhibition among the excitatory neurons. Chapter 4 implements this V2 variant in fixed-point programmable logic, reports measured accuracy, latency, resources, power, and energy, and uses that measurement to interpret the potential hardware effect of the preceding algorithmic controls. Chapter 5 summarizes the results and identifies clinical biosignal validation, combined-controller experiments, and implantable-ASIC realization as future work.

These chapters share the input encoding, adaptive excitatory dynamics, STDP, normalization, labeled readout, and the W–θ–r state decomposition; they do not all use an identical inhibitory topology. Their validation boundaries must therefore remain explicit. Chapters 2 and 3 are algorithmic studies. Chapter 4 implements the V2 baseline rather than the complete set of selective-adaptation and imbalance controls. The measured FPGA cost can therefore be used to project where those controls would save work, but such projections are not measurements of a combined accelerator. Likewise, the FPGA is a development platform, not an implantable ASIC, and MNIST is a controlled benchmark, not a clinical recording.

Within those boundaries, the thesis establishes a consistent design principle: plasticity is a resource that should be allocated with the same care as computation and memory. The reliability of the available learning signal should determine which state is writable; the statistics of the neuron population should determine which representations require protection; and hardware evaluation should count the state transactions produced by those decisions. The following chapters develop that principle algorithmically and then measure the baseline on hardware.

Table 1.3: Parameters of the original Diehl–Cook/BindsNET baseline used for the original-model experiments in this thesis. Derived per-step decay factors connect the implementation with the equations in this chapter.

Group	Parameter	Baseline setting	Interpretation and exceptions
Architecture and input encoding			
	M	784	One input channel per 28×28 MNIST pixel.
	N	400	Number of adaptive excitatory neurons.
	N_{I}	400	Number of paired inhibitory neurons.
	C	10 or 2	Ten-class MNIST baseline; $C = 2$ after one-versus-rest relabeling in the rare-event studies.
	Δt_s	1 ms	Simulation timestep.
	T	250 timesteps	One sample is presented for 250 ms.
	γ	128 Hz	Maximum Poisson rate after scaling a normalized pixel intensity.
Excitatory-neuron dynamics			
	$v_{\mathrm{rest}}, v_{\mathrm{reset}}$	$-65, -60$ mV	Resting and post-spike reset potentials.
	v_{thr}	-52 mV	Base threshold; the instantaneous threshold is $v_{\mathrm{thr}} + \theta_j$.
	τ_m, β	100 ms, 0.99005	$\beta = \exp(-\Delta t_s/\tau_m)$.
	H_{ref}	5 timesteps	Post-spike refractory duration, equal to 5 ms.
	θ_+	0.05 mV	Adaptive-threshold increment after an output spike.
	τ_θ, ρ_θ	10^7 ms, 0.9999999	$\rho_\theta = \exp(-\Delta t_s/\tau_\theta)$; θ persists between samples.
	one-spike rule	enabled	At most one output neuron is selected at each timestep.
Inhibitory population and lateral competition			
	$v^{\mathrm{I}}_{\mathrm{rest}}, v^{\mathrm{I}}_{\mathrm{reset}}, v^{\mathrm{I}}_{\mathrm{thr}}$	$-60, -45, -40$ mV	Rest, reset, and firing threshold of each inhibitory neuron.
	$\tau^{\mathrm{I}}_m, H^{\mathrm{I}}_{\mathrm{ref}}$	10 ms, 2 timesteps	Inhibitory-neuron membrane time constant and refractory duration.
	$g_{\mathrm{E}\to\mathrm{I}}$	22.5	Fixed one-to-one excitation from each excitatory neuron to its paired inhibitory neuron.
	$g_{\mathrm{I}\to\mathrm{E}}$	120	Magnitude of the fixed inhibitory connection to every non-paired excitatory neuron.
STDP, normalization, and readout			
	$\tau_{\mathrm{pre}}, \tau_{\mathrm{post}}$	20, 20 ms	Both trace decays are $\lambda = \exp(-1/20) \approx 0.95123$ per timestep.
	η_-, η_+	$10^{-4}, 10^{-2}$	Depression and potentiation rates in the PostPre rule.
	$w_{ij}[0]$	$\mathcal{U}[0, 0.3)$	Independent initial feedforward weights.
	$w_{\min}, w_{\max}$	0, 1	Feedforward-weight clipping bounds.
	κ	$78.4 = M/10$	Target L_1 sum of each neuron's incoming weight vector.
	T_{label}	250 samples	Class rates, proportions, and assignments are refreshed at this interval in the BindsNET experiments.
	α	1	No decay of earlier class-response evidence in the baseline; later policies state any different update routing.

Table 1.4: Interpretation and limits of the controlled validation environment.

Target-system concern	Controlled proxy	What the proxy does not establish
Rare pathological event amid varied background	One digit versus the remaining nine, with adjustable imbalance	Clinical prevalence, artifacts, or event morphology
Patient or session variation	Rotation, translation, and zoom applied after pretraining	A physiological model of inter-patient or longitudinal change
Limited deployment labels	Sequential adaptation stream without ground-truth input to the controller	Safety of pseudo-labels in a clinical decision path
Local resource constraints	Fixed-point FPGA inference and STDP learning	Implantable-ASIC area, thermal safety, packaging, or battery lifetime

2 Confidence-Based Selective Adaptation

Chapter 1 introduced the baseline excitatory–inhibitory STDP-SNN and its three persistent plastic states: the synaptic weight matrix W, the adaptive thresholds θ, and the firing-rate statistics r used for neuron-to-class assignment. A conventional implementation leaves these states plastic throughout training and applies their respective update rules whenever their update conditions are met. This uniform treatment is convenient algorithmically, but it is poorly matched to continuous deployment. The three states differ in their function, convergence behaviour, size, and cost of modification. In particular, W stores a complete receptive field for every excitatory neuron and is much larger than the two neuron-level states. Writing it indiscriminately can both disturb a useful representation and dominate the memory traffic of online learning.

This chapter asks three narrower questions that follow from the background in Chapter 1. First, do the three states remain equally useful as learning proceeds? Second, in the absence of deployment labels, can an internally generated signal determine which state should adapt to a particular sample? Third, after a sample has been routed to the weight path, do all active neurons require another expensive weight commit? The answers lead to two mechanisms. A confidence-conditioned dual-path rule routes reliable samples to slow weight consolidation and unreliable samples to the smaller homeostatic and assignment states. A sample-relative selectivity rule then retains weight plasticity only for the less-selective neurons among those involved in the current decision. The chapter develops these mechanisms from ablations rather than assuming the update policy in advance, and evaluates both task performance and the resulting weight-memory activity.

A conference version of the core method and experiments developed in this chapter has been accepted for publication [53].

2.1 Stability–Plasticity and Selective Learning

Continuous adaptation poses a stability–plasticity problem. A network must remain plastic enough to follow a changed input distribution, yet stable enough to preserve features that remain useful. Repeated learning on a new stream can otherwise overwrite earlier knowledge, a failure commonly described as catastrophic forgetting. Multiple-state models address the underlying tension by separating rapidly changing and persistent variables. The cascade model places a synapse in hidden states of different stability [54], whereas the coupled-variable model transfers information between fast and slow processes to extend memory lifetime [55]. These models establish the general principle that plastic variables need not share a single timescale,

but they do not prescribe how the distinct states of a deployed classifier should be routed for each input.

Importance-based continual-learning methods protect parameters that support previous tasks. Elastic weight consolidation estimates importance from the Fisher information and penalizes later movement of important weights [33]. Synaptic intelligence instead accumulates an online estimate of each parameter's contribution to changes in the training objective [56]. These methods provide a useful analogy for protecting a specialized SNN neuron. They nevertheless depend on gradient-based objectives and normally consolidate at identified task transitions. The mechanism developed here operates without a deployment loss, backward pass, or task identifier and protects the incoming weight row of a neuron using class selectivity already available in the SNN readout.

A closer spiking precedent is the Controlled Forgetting Network of Allred and Roy [36]. When the ordinary excitatory population does not respond to an input, an additional dopaminergic neuron treats the sample as novel, stimulates a selected neuron, and temporarily increases its plasticity so that new information is learned without exposing the full population to interference. In a completely disjoint sequential MNIST experiment, the 6,400-neuron network retains 95.24% accuracy, compared with 32.97% without dopaminergic modulation or homeostasis and 61.95% with homeostasis alone. The method is unsupervised during learning and uses labels only for the fixed neuron-to-class assignment and evaluation. It targets class-incremental novelty through an additional detection and stimulation pathway, whereas the present method uses the existing population confidence and selectivity statistics to route state updates during covariate shift.

Conditional plasticity also has direct precedents in spiking networks. Brader, Senn, and Fusi introduced a spike-driven bistable-synapse rule in which postsynaptic depolarization and a calcium-like firing-rate variable establish potentiation, depression, and stop-learning regions [57]. Their instructor-driven system demonstrates that learning can be suspended dynamically once an output becomes satisfactory. Noack et al. subsequently implemented 8,192 such stop-learning synapses in a 28-nm switched-capacitor system, with an externally computed calcium state controlling stop bits [58]. Thus, a no-update state is not itself new. The distinction in this chapter is the decision signal: prediction confidence determines which plastic states are updated, and a neuron's learned class selectivity determines whether its entire incoming row remains eligible for commitment.

Other SNN approaches soften plasticity rather than switching it categorically. Adaptive Synaptic Plasticity combines STDP with activity-dependent weight decay to retain significant features while gradually forgetting insignificant ones [35]. Antonov et al. show that spiking representations, lateral inhibition, and local

STDP/R-STDP rules do not alone prevent interference; they estimate permissible weight ranges through Langevin dynamics and compare this protection with replay and other strategies [37]. Naderi et al. take a phase-based approach: after long-term Triplet-STDP training, the weights are frozen while short-term synaptic efficacy remains dynamic [50]. These works motivate separating stable and adaptive states, but none performs the sample-wise routing among W, θ, and r developed below.

The signal that authorizes plasticity is equally important. Three-factor learning extends a pre/post Hebbian event with a modulatory quantity that can represent reward, novelty, or surprise [59]. A classical example links an STDP eligibility trace with delayed dopamine reinforcement [60]; more recently, Barry and Gerstner derived an endogenous surprise signal from excitation–inhibition mismatch and used it to increase plasticity after a predictive rule changed [61]. Confidence has also been used to reject unreliable pseudo-labels during gradual SNN domain adaptation, although that method uses two alternating teacher–student networks, error-modulated STDP, and a feedback path [38]. ETLP similarly uses teaching events and projected labels as a third factor and update trigger [62]. These methods establish that plasticity may be conditional, but their supervisory requirements and update targets differ from the label-free deployment rule considered here.

A particularly relevant recent method is Astrocyte-Gated Multi-Timescale Plasticity (AGMP), which combines eligibility traces, homeostasis, a broadcast teaching/error signal, and a slow activity-derived gate. The gate suppresses weight updates in stable regimes and reopens them as activity changes [63]. AGMP therefore shares this chapter's use of multiple timescales and heterogeneous SNN plasticity. It remains supervised, however, and applies its gate to a common eligibility-based weight rule; it does not use confidence bands to route separate synaptic, homeostatic, and assignment states.

Hardware work provides the final part of the gap. Existing processors reduce different parts of online-learning execution. Li et al. combine adaptive clock/event-driven scheduling with neighboring-PE borrowing, whereas Sun et al. combine temporal coding and simplified STDP with termination of the forward pass after a decisive spike [64, 65]. TripleBrain instead provides a compact event-driven core configurable for unsupervised STDP and supervised R-STDP variants, providing accuracy–execution-cost trade-offs through the choice and duration of the learning rule [66]. These processors reduce learning duration or the work performed during learning; they do not use the current sample to decide whether a persistent update is necessary. Probabilistic metaplasticity addresses the write decision more directly by making the programming probability of each memristive SNN weight depend on its importance [67]. It nevertheless uses target spike trains and local error neurons and gates individual weight updates under supervision, rather than routing

label-free updates among synaptic, homeostatic, and assignment states. The gap addressed in this chapter is consequently not any individual ingredient, but their organization as a state- and write-aware deployment policy.

2.2 Plastic-State Ablation and Timescale Separation

The baseline network and its learning equations were introduced in Section 1.5. The purpose of the first experiment is not to assign the states to confidence regions, but to determine how long each state remains necessary after the initial representation has formed. The quantities that are independently frozen are summarized in Table 2.1. No proposed update path is included in the table, because that routing is a result of the later confidence-conditioned ablation.

Table 2.1: Persistent plastic states independently controlled in the timescale ablation. The table describes the baseline network only; confidence-conditioned update regions are derived later in Section 2.4.

State	Size	Update driver	Function
W	$N_{\text{pre}}N_{\text{post}}$	pre/post spike timing	receptive fields
θ	N_{post}	output spikes and decay	firing-rate homeostasis
r	$N_{\text{post}}C$	class-indexed spike counts	assignment and readout

The asymmetry is substantial. W contains $N_{\text{pre}}N_{\text{post}}$ synapses. For every postsynaptic neuron that spikes, STDP modifies its incoming vector, which is subsequently normalized. By contrast, θ contains one scalar per excitatory neuron, and r holds only C accumulated class responses per neuron. Size alone, however, does not establish that W can be updated less often; that requires observing the network after learning has begun. This ablation uses the same binary one-versus-rest construction as the rare-event benchmark, so $C = 2$ within each run. One MNIST digit is relabelled as the positive event class and all nine remaining digits are pooled into the negative background class. Sensitivity is therefore recall of the selected digit, whereas specificity is recall of the pooled background. The complete study repeats this two-class experiment with each of the ten digits serving as the event class in turn and averages the results over the repeated runs, as detailed in Section 2.7.

The remaining contribution of each state is tested with a freeze-after-epoch-1 experiment. All configurations learn normally during the first epoch of a ten-epoch pretraining run. Beginning with epoch 2, one configuration freezes W, a second freezes θ, and a third freezes r, while the other two states remain plastic. The

intervention therefore asks for the remaining contribution of each state after the first pass through the data. It is not an online policy and does not use confidence.

The result in Figure 2.1 is more informative when read in absolute terms. Freezing W retains 93% accuracy, 92% specificity, and 95% sensitivity, essentially matching the fully plastic reference. Freezing r reduces accuracy to 88%, whereas freezing θ reduces it to 80% and specificity to 78%. The high sensitivity obtained when θ is frozen does not compensate for the accompanying loss of pooled-background recall: the threshold is needed to keep firing competition balanced. Likewise, the loss after freezing r shows that class evidence and neuron assignments are still being resolved after the receptive fields have largely formed.

The experiment therefore identifies an effective separation of timescales in this network. Early STDP and normalization establish useful receptive fields quickly; further movement of W adds little on the stationary pretraining distribution. Threshold homeostasis and class-response statistics remain consequential for performance. This does not establish where any of these states should update under a shifted deployment stream. It only motivates the next question: which samples provide a sufficiently reliable signal for the slow, expensive state, and which samples should be used for the faster states? The interpretation is consistent with the broader multi-timescale consolidation principle [54, 55], but here it is obtained directly from the state ablation rather than imposed as a model assumption.

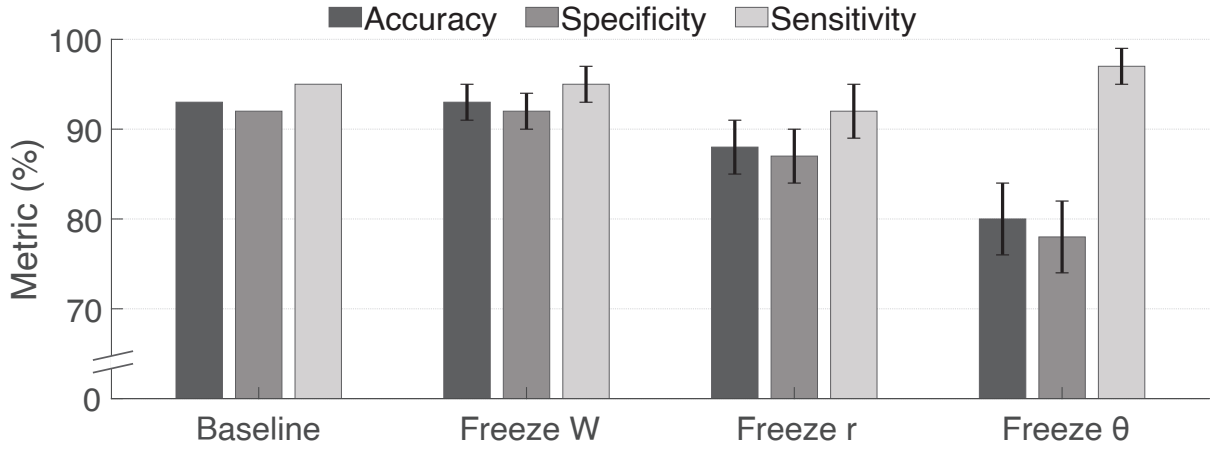


Figure 2.1: Freeze-after-epoch-1 ablation during binary one-versus-rest pretraining. Within each run, sensitivity refers to the selected event digit and specificity to the nine digits pooled into the background class. Freezing the synaptic weights W after the initial epoch has little effect, whereas freezing the adaptive thresholds θ or firing-rate tracker r degrades performance. Error bars report standard deviation over the repeated experiments.

2.3 Confidence as an Internal Supervision Signal

The timescale ablation shows that the states are no longer equally useful after the initial epoch, but it does not identify a sample-wise supervisory signal. At deployment, the ground-truth class is unavailable. The signal must therefore be computed from the same output spikes and accumulated class statistics already used by the readout.

Let $r_{j,k}$ be the accumulated response of excitatory neuron j to class k. Its normalized class proportions $p_{j,k}$ and current assignment a_j are

$$p_{j,k} = \frac{r_{j,k}}{\sum_{k'=1}^{C} r_{j,k'}}, \qquad a_j = \arg\max_k p_{j,k}. \tag{17}$$

For a presented sample x, let $s_j(x)$ be the number of spikes emitted by neuron j. Each spike contributes the corresponding proportion vector $p_{j,:}$ to the population vote. The resulting class score is

$$q_k(x) = \sum_{j=1}^{N_{\text{post}}} s_j(x) p_{j,k}. \tag{18}$$

The predicted class and confidence are then

$$\widehat{y}(x) = \arg\max_k q_k(x), \qquad c(x) = \frac{q_{\widehat{y}(x)}(x)}{\sum_{k=1}^{C} q_k(x)}. \tag{19}$$

Because every proportion vector sums to one, the denominator is also the total number of output spikes. Confidence is therefore the fraction of the weighted population vote received by the winning class, rather than the maximum response of one neuron. For a nonzero population response in the binary rare-event task, $c \in [0.5, 1]$: $c \approx 0.5$ means that the population vote is split, whereas $c \approx 1$ means that nearly all weighted activity supports the same class.

This construction separates confidence from raw firing strength. Scaling the same population response to a larger total spike count scales all class scores but leaves their normalized proportions unchanged. Confidence becomes high only when the active population has a clear class consensus. It is also inexpensive to obtain in an event-driven implementation: on every output spike, the readout accumulates one stored proportion row into the C class scores, followed by one normalization at the sample boundary.

Confidence is useful as a learning signal only if it correlates with prediction correctness. Figure 2.2 bins a held-out post-pretraining test set by c and separates correct from incorrect decisions using the labels only for

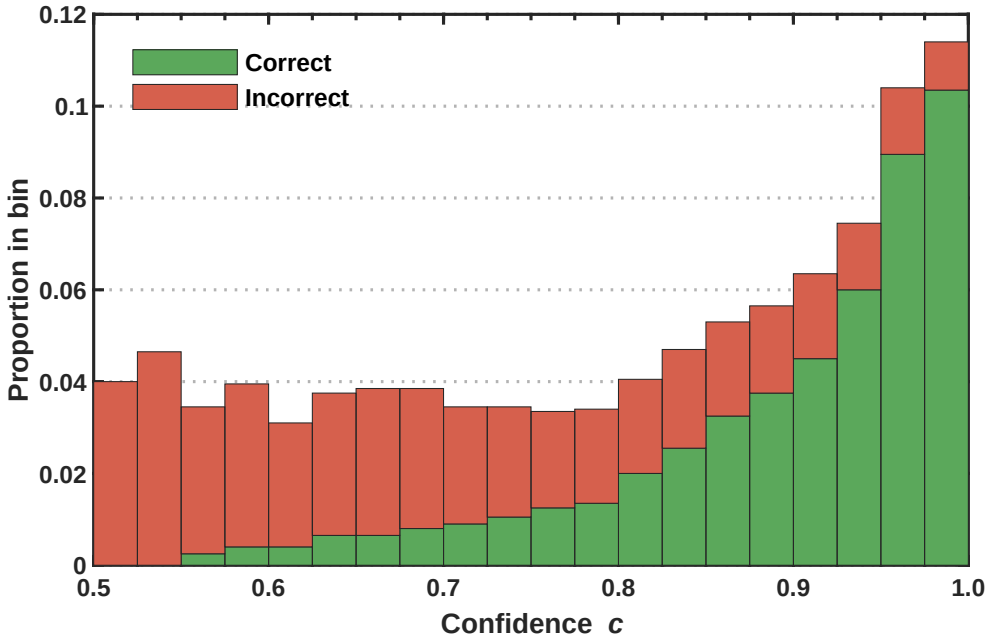


Figure 2.2: Distribution of correct and incorrect predictions across confidence bins on a held-out post-pretraining test set. Labels are used only to evaluate correctness in this diagnostic. Correct decisions become more prevalent as c increases, supporting confidence as a label-free reliability signal for update routing.

this offline diagnostic; the labels do not enter the confidence calculation. Incorrect predictions dominate the lowest-confidence region, whereas correct predictions increasingly outnumber incorrect predictions as confidence approaches one. No shifted calibration set is reported for this diagnostic. The plot should therefore be interpreted as an empirical correlation on the held-out post-pretraining set, not as evidence that confidence remains calibrated under shift, where confidently wrong decisions can occur.

The confidence-conditioned ablation uses two operating points read from this curve: $c_{\text{low}} = 0.60$ and $c_{\text{high}} = 0.90$. They divide the stream into a high-confidence subset, a low-confidence subset, and an intermediate band. At this stage, the thresholds only define the subsets on which the next ablation is performed; they do not yet specify which state is allowed to update. The intermediate band is kept separate so that the two diagnostic populations have clearly different reliability, rather than mixing borderline samples into either group.

2.4 Confidence-Conditioned Ablation and the Dual-Path Rule

Confidence specifies when the prediction is reliable, but it does not by itself identify which state should update. This question is resolved with a second ablation. The online data are divided into confident ($c \geq c_{\text{high}}$) and non-confident ($c \leq c_{\text{low}}$) sub-streams. Within each sub-stream, one state is frozen while the other two

continue updating. Each bar therefore measures the performance change caused by removing one state from an otherwise all-plastic path. Ground-truth labels are used in this diagnostic to remove pseudo-label error as a confounding factor. They are not inputs to the deployment policy derived from the experiment.

Figure 2.3 reports the change relative to the all-state configuration for a representative digit-3-versus-background rotation case. On the high-confidence path, freezing W causes a 33.8-percentage-point accuracy loss, whereas freezing r reduces accuracy and sensitivity by only 1.9 points. Freezing θ leaves accuracy unchanged and improves sensitivity by 29.9 points. Thus, W carries the useful adaptation on reliable samples, while continuing threshold adaptation on this majority-dominated path can suppress minority participation.

The low-confidence path has a different hierarchy. Freezing r reduces accuracy and sensitivity by 21.6 and 26.2 points, respectively, and freezing θ reduces them by 12.9 and 23.4 points. Freezing W reduces accuracy by 8.7 points but sensitivity by only 1.9 points. The accuracy effect is not negligible, although it is smaller than the effect of freezing θ or r; its sensitivity effect is much smaller. The data therefore do not show that W is intrinsically harmful on every uncertain sample. Rather, they show that θ and r contain the dominant corrective value for the minority class, while keeping W plastic on that path produces little sensitivity benefit at the cost of committing the largest state. The minimal low-confidence path consequently keeps only θ and r plastic and freezes W as an explicit performance–write-cost trade-off.

The resulting update policy is

$$\mathcal{U}(c) = \begin{cases} \{W\}, & c \geq c_{\text{high}}, \\ \{\theta, r\}, & c \leq c_{\text{low}}, \\ \varnothing, & c_{\text{low}} < c < c_{\text{high}}. \end{cases} \tag{20}$$

On the confident path, the baseline STDP rule remains enabled for the postsynaptic neurons that spike; it is not restricted to the single readout winner. Their incoming weight vectors form the candidate commits that will be refined by the selectivity rule in Section 2.5. θ and r are held fixed on this path. On the non-confident path, W is held fixed while the active-neuron thresholds and class-response statistics are updated. The current prediction $\hat{y}$ supplies the class index used to update r. Samples in the intermediate band perform inference only.

This routing is more specific than filtering pseudo-labels. Low-confidence samples are not trusted to modify the receptive fields, but they are not discarded: their activity is used by the two smaller states that regulate competition and maintain the readout. High-confidence samples provide the coherent pre/post activity used

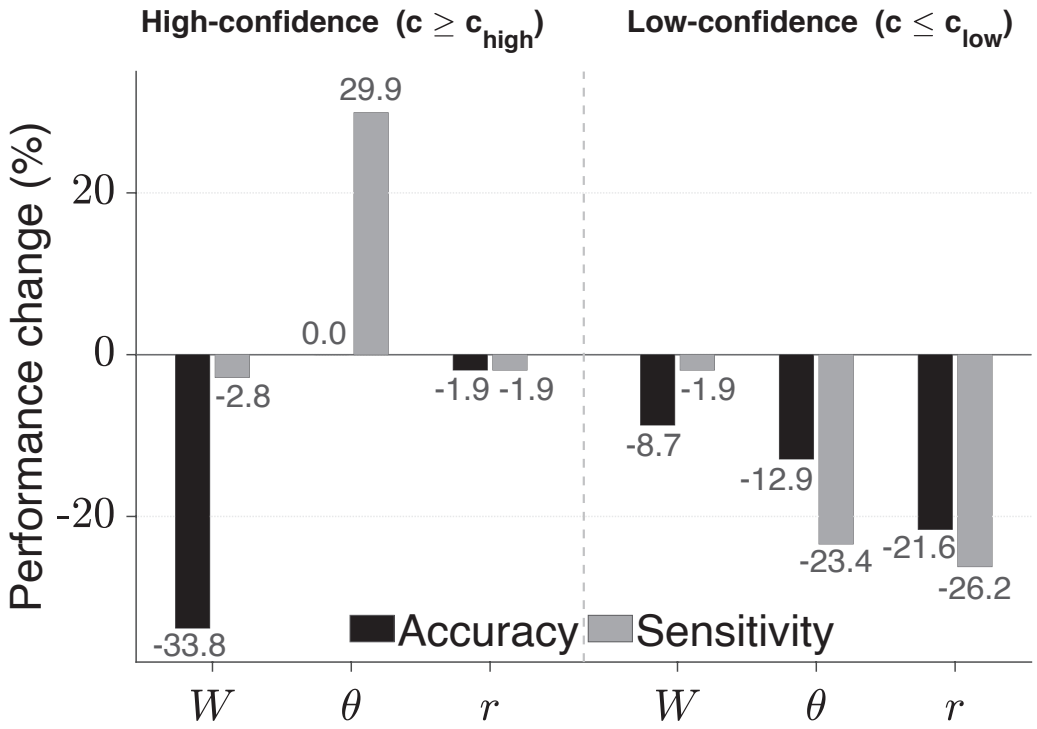


Figure 2.3: Confidence-conditioned state ablation. Each bar is the performance change caused by freezing one state relative to the all-state configuration on that path. The confident and non-confident regimes have different state requirements, motivating the dual-path rule.

for synaptic consolidation without continuing the threshold changes shown to hurt minority sensitivity.

This organization can be viewed as a network-internal three-factor rule. The pre/post spike relationship still supplies the candidate synaptic change, while confidence controls whether it is committed. The analogy is functional rather than biological: confidence is not asserted to be a dopamine or reward signal. More importantly, the gate controls an entire state class rather than merely scaling one weight update.

2.5 Sample-Relative Selectivity Gating

The dual-path rule reduces how many samples reach W, but all spiking neurons on a high-confidence sample would still be eligible for STDP and normalization. This is visible in the measured update pressure: a sample can produce more than one weight-vector commit even though the readout returns one class. The second gate therefore operates on the set of neurons involved in the current sample, not only on the readout winner.

The class selectivity of neuron j is the margin between the accumulated proportion of its preferred class a_j

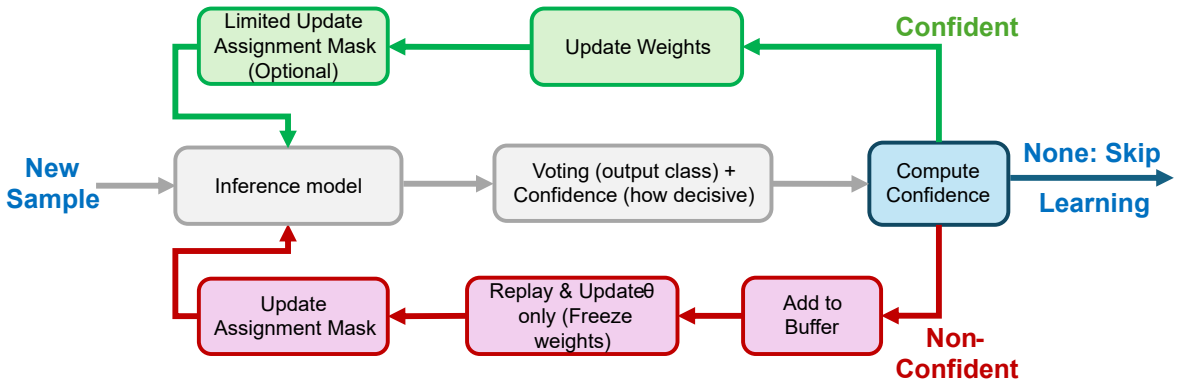


Figure 2.4: Control flow of the confidence-conditioned dual-path policy. The network first performs inference and forms normalized class votes. High- and low-confidence samples enable different plastic states, while the intermediate band performs inference without a state update. The high-confidence branch produces candidate weight commits; the sample-relative selectivity refinement is introduced separately in Section 2.5.

and the total proportion assigned to all other classes,

$$\sigma_j = p_{j,a_j} - \sum_{k \neq a_j} p_{j,k} = 2p_{j,a_j} - 1. \tag{21}$$

The second equality follows because the class proportions sum to one. In the binary experiments, the sum contains only the other class, so $\sigma_j = p_{j,a_j} - p_{j,\text{other}}$. Hence $\sigma_j \in [0,1]$: a value near zero means that the neuron's accumulated response is divided almost equally between the event and background classes, whereas a value near one means that its preferred class accounts for nearly all of its response. The method does not compare σ_j with a fixed selectivity threshold. For each sample x, it first forms the active set

$$\mathcal{A}(x) = \{j : s_j(x) > 0\} \tag{22}$$

and computes the mean selectivity of the neurons in that sample,

$$\overline{\sigma}(x) = \frac{1}{|\mathcal{A}(x)|} \sum_{j \in \mathcal{A}(x)} \sigma_j. \tag{23}$$

On a high-confidence sample, the incoming weight vector of an active neuron is committed only when its selectivity is not greater than the sample mean:

$$j \in \mathcal{G}(x) \quad \Longleftrightarrow \quad c(x) \geq c_{\text{high}},\ j \in \mathcal{A}(x),\ \sigma_j \leq \overline{\sigma}(x). \tag{24}$$

Neurons above the sample mean still participate in inference and lateral competition, but their candidate weight commits are suppressed for that sample.

On a high-confidence input, the above-mean group represents the relatively class-selective part of the active population. Suppressing its weight commits leaves these neurons' receptive fields unchanged on that input, while allowing them to continue participating in inference and lateral competition. Across high-confidence inputs, this preferential protection reduces changes to the more selective receptive fields, while the less-selective neurons remain plastic and can absorb changes in the input distribution. The protected neurons can therefore act as relatively stable anchors for population voting, providing established class evidence while the rest of the active population adapts.

The distinction from a global threshold is important. The rule is relative to the population that participated in the current decision and introduces no selectivity hyperparameter. It preferentially spends writes on the active neurons whose class evidence is less resolved than that of their peers, while avoiding another commit to the comparatively class-specific neurons in the same sample. The comparison is recomputed from the current $p_{j,k}$ values on every sample; there is no persistent stop bit and no claim that crossing a fixed value proves convergence. All required quantities are already present in the readout, so the refinement adds comparison and accumulation logic but no new learned state. Its empirical justification is consequently the result reported later: reduced weight-update pressure without a loss of minority-class sensitivity.

2.6 Labelling and Supervision

The distinction between diagnostic labels, initialization labels, and deployment supervision is essential. The baseline performs unsupervised STDP feature learning, but class labels are used during offline pretraining and readout calibration to accumulate r and establish the initial neuron-to-class association, as in [44]. True labels are also used in the confidence-conditioned ablation solely to isolate the contribution of each state. Neither use implies that labels are available to the deployed update controller.

During deployment, each state is updated from the following signals:

- W uses only local pre- and postsynaptic spike timing and is enabled only by the confidence and selectivity conditions;
- θ depends only on output-neuron activity and its homeostatic decay;

- r uses the network's population-vote prediction $\widehat{y}(x)$ as a pseudo-label for the active neurons' class-response statistics.

No ground-truth class, reward, global loss, backward pass, teacher network, or explicit shift boundary is required online. This distinguishes the method from supervised R-STDP systems [68, 66], learning with projected teaching labels [62], and teacher–student SNN adaptation with error-modulated STDP and feedback [38]. All principal evaluations in Section 2.7 use this label-free deployment configuration.

Pseudo-labelling nevertheless introduces a limitation. Confidence reduces but does not eliminate the risk of confirmation bias. A confidently wrong prediction can still authorize a harmful weight commit, and a low-confidence sample can arise from noise, ambiguity, or an unseen class rather than a benign covariate shift. The high-confidence threshold and intermediate abstention band make the policy conservative, while protecting W on the low-confidence path bounds the most expensive form of error reinforcement. These mechanisms are safeguards rather than a guarantee of correct self-supervision.

2.7 Results

2.7.1 Evaluation protocol

A controlled benchmark is used to vary imbalance and shift independently. One MNIST digit is designated as a rare-event class and the other nine digits are combined into a background class. The online stream uses a 1:10 minority-to-background ratio. Each digit is selected as the minority class in turn, and each condition is repeated over five random seeds; reported values are averages with standard-deviation error bars. This construction is not a claim of clinical validation. It is a reproducible proxy for studying the learning-cost/performance trade-off under rare-event conditions. Chapter 3 examines class imbalance itself in greater depth.

The network contains 400 excitatory and 400 inhibitory neurons. For each choice of event digit, it is pre-trained on 60,000 MNIST samples after the binary one-versus-rest relabelling described above, using the baseline implementation. The deployment evaluation uses unlabelled rare-event streams with one of three uniformly sampled covariate shifts: rotation over $\pm 60^{\circ}$, translation over ± 3 pixels, or zoom over $\pm 40\%$. Task performance is evaluated over 10,000-sample online streams. The longer update-pressure trace in Figure 2.6 uses 60,000 samples to expose both the transient and steady-state regimes. The compared configurations are no adaptation, naive all-state adaptation, the confidence-gated dual path, and the dual path with

selectivity gating. Minority-class sensitivity is the primary metric; specificity and accuracy are also reported.

2.7.2 Robustness under covariate shift

Figure 2.5 compares no adaptation, naive all-state adaptation, and the dual-path rule. The three metrics must be read together because the imbalanced stream allows an improvement in minority recognition to be hidden by the much larger background class.

Naive adaptation changes the fixed-model result very little. Its sensitivity gain is only 1.0 percentage point in each shift, while its accuracy and specificity differ by at most 0.2 points. Thus, merely leaving all three states plastic does not extract much benefit from the unlabelled stream. This result is consistent with the confidence-conditioned ablation: updates driven by reliable and unreliable samples can oppose one another, so useful consolidation can be offset by changes made from uncertain samples and by unnecessary homeostatic updates on already-confident background examples.

The dual-path rule improves the mean minority sensitivity relative to no adaptation by 2.2 points under rotation, 5.6 points under translation, and 4.6 points under zoom. Relative to naive adaptation, the corresponding gains are 1.2, 4.6, and 3.6 points. Translation therefore shows the largest overall benefit: dual-path routing also raises accuracy by 4.5 points and specificity by 4.5 points relative to no adaptation. Under rotation, it raises accuracy and specificity by 3.9 and 4.1 points, respectively. These two cases show that the sensitivity gain is not obtained by indiscriminately predicting the event class; background rejection improves at the same time.

Zoom gives a different trade-off. Dual-path adaptation raises sensitivity from 79.8% to 84.4%, but its mean accuracy remains at 85.5% and specificity falls from 86.0% to 85.5%. The 0.5-point specificity change is small compared with the reported variability, but it means that the result should be described as approximately preserving overall performance, rather than as never degrading any mean metric. This distinction matters under the 1:10 imbalance because accuracy is dominated by the background class and can conceal a materially different minority sensitivity.

Finally, the error bars report standard deviation and overlap in several comparisons. No significance test is reported, so the figure supports a descriptive conclusion about the ordering of the means, not a claim of statistical significance for every pairwise difference. The strongest result is consequently the consistent direction of the sensitivity change across all three shift types, together with the much larger reduction in weight-update pressure quantified next.

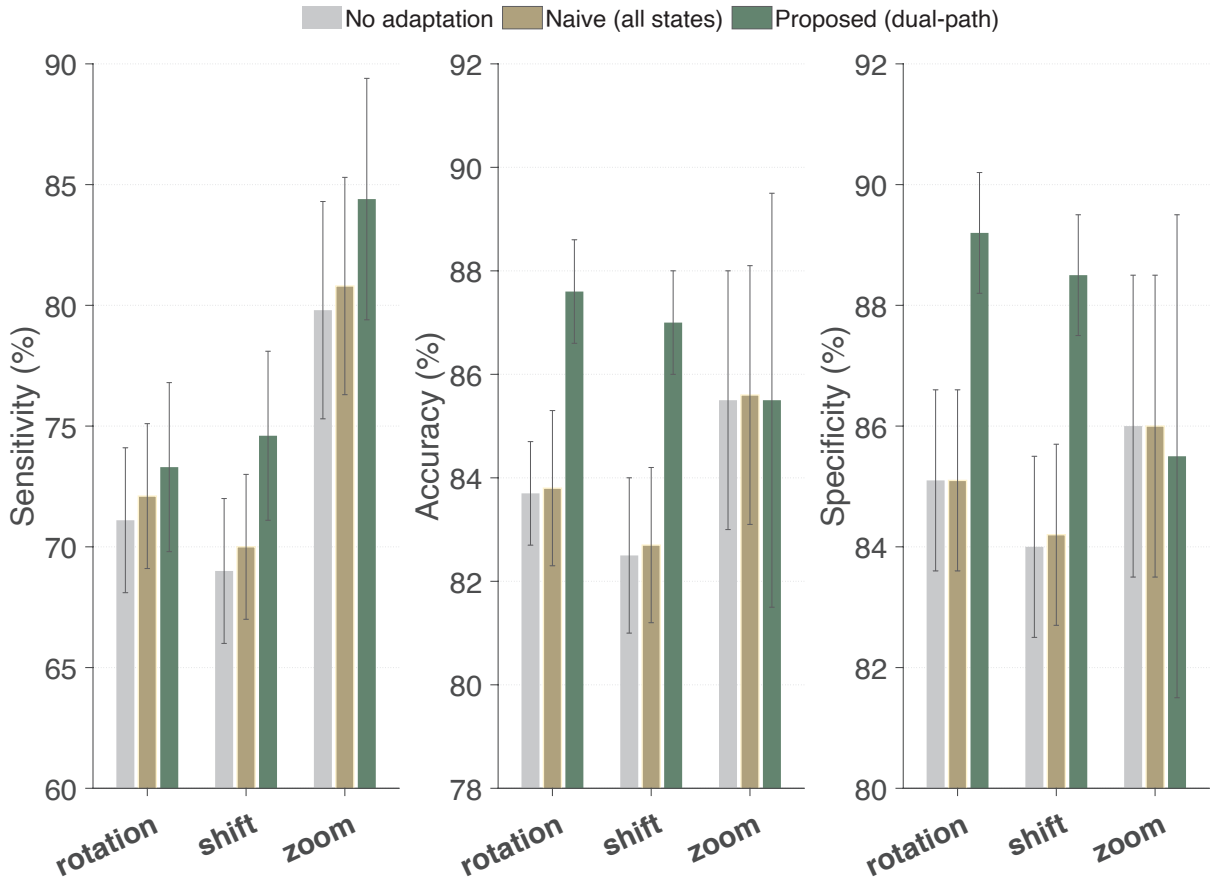


Figure 2.5: Minority-class sensitivity, accuracy, and specificity under rotation, translation, and zoom shifts. The confidence-gated dual-path rule provides the highest mean sensitivity in each regime. Accuracy and specificity improve under rotation and translation; under zoom, the sensitivity gain is accompanied by a small decrease in mean specificity. Error bars report standard deviation over the repeated experiments.

2.7.3 Weight-update pressure

The hardware-relevant quantity is update pressure: the expected number of neurons whose complete incoming weight rows are committed per input sample. It captures how frequently the confidence path is entered and how many candidate rows survive the sample-relative comparison. Figure 2.6 tracks this quantity for naive adaptation, the dual-path rule, and the dual path with selectivity gating.

During active adaptation, confidence routing reduces update pressure by approximately 65% relative to naive adaptation. Adding the selectivity gate increases the reduction to approximately 80% without changing minority-class sensitivity. The additional saving is largest early in adaptation, when many neurons are already selective but the ungated dual path would continue recommitting their rows. As the network settles, both gated configurations approach a steady state approximately 92% below naive adaptation. What narrows

over time is the incremental gap between the two gated traces, not the saving relative to continuous all-state STDP.

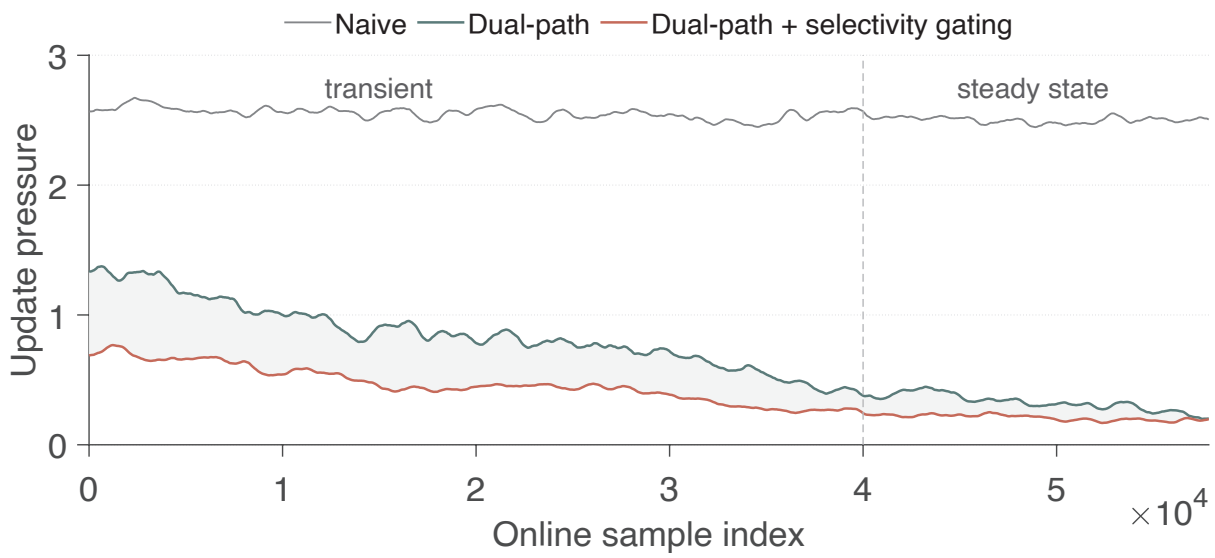


Figure 2.6: Update pressure over a 60,000-sample online run. Confidence routing removes most weight commits, while the sample-relative selectivity comparison suppresses additional redundant commits during the early active-adaptation phase. Both gated configurations settle far below the naive all-state baseline.

2.8 Hardware Implications and Scope of the Estimate

A weight commit reads, normalizes, and writes one postsynaptic neuron's N_{pre} incoming synapses. It therefore requires approximately N_{pre} reads and N_{pre} writes to the largest state store. By contrast, a θ or r update touches neuron-level entries. Using the published 45-nm, 0.9-V, 8-bit access values in [41], the approximately 314-kB W memory in the 784×400 network with 8-bit weights is modelled at about 50 pJ per access. A complete weight-row commit is therefore approximately 78 nJ per neuron. The combined threshold/rate update is approximately 60 pJ per neuron, making one row commit roughly 1,300 times more expensive in this access model.

Figure 2.7 applies the same model to measured update activity. Naive all-state adaptation averages 2.59 weight commits and approximately 202 nJ of weight-update memory energy per sample. Confidence routing reduces the averages to 0.52 commits and 41 nJ. Adding selectivity gating reduces them to 0.21 commits and 16 nJ. Activity on the non-confident θ and r path is visible in the component count but contributes negligibly to the energy total under these assumptions. The principal saving therefore comes from preventing large-row memory transactions, not from optimizing arithmetic on the small states.

The reductions reported by these hardware studies use different denominators and baselines, so they should

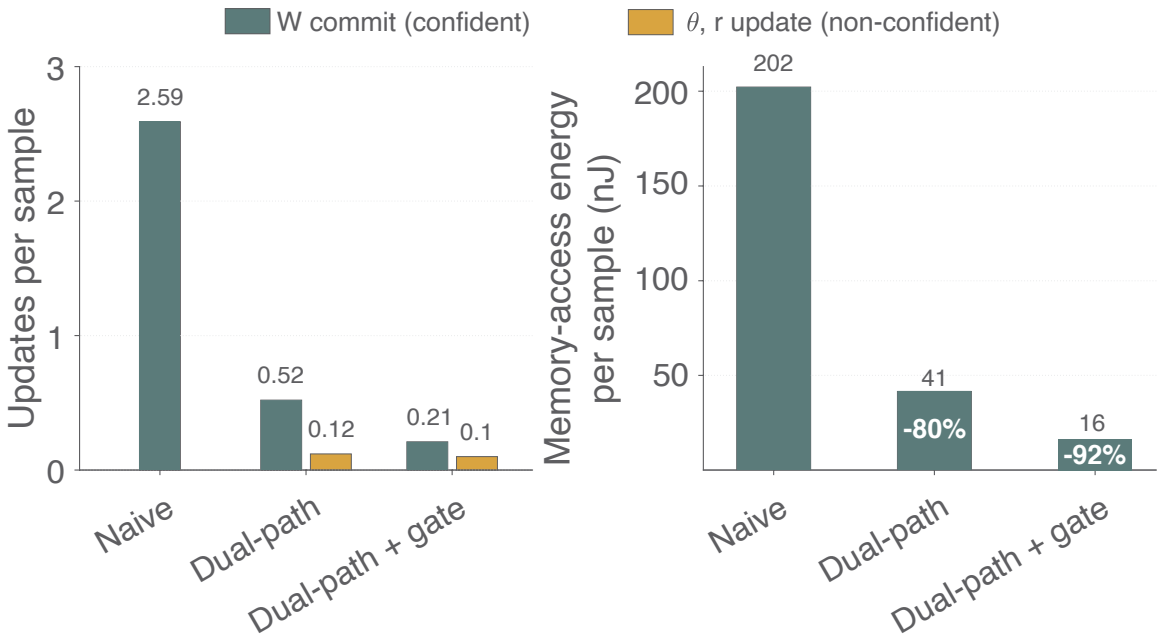


Figure 2.7: Estimated on-chip learning cost. (a) Average update activity for weight commits and non-confident θ and *r* updates. (b) Weight-update memory energy obtained by applying a published 45-nm access model to the measured activity. These values are analytical estimates, not FPGA or silicon power measurements of the proposed controller.

not be read as a direct ranking against Figure 2.6. In [64], enabling adaptive execution and neighboring-PE borrowing is reported to reduce online-learning time by approximately 24% and estimated energy by 20% in a within-design comparison. In [65], the stop mechanism is reported to reduce the average number of evaluated input pixels during inference from approximately 550 to 120, or by about 78%; this is a reduction in forward-pass computation, not a measure of avoided weight writes. In [66], stopping the highest-accuracy learning schedule after the third rather than the seventeenth epoch is reported to reduce learning latency by about 85%, but to lower MNIST accuracy by 1.89 percentage points. The matched 65-nm mixed-signal analysis on split-MNIST in [67] reports up to three orders of magnitude fewer update-eligible weights and up to 60$\times$ lower parameter-update energy than activity-dependent metaplasticity with gradient accumulation. That comparison uses the same 200-hidden-neuron continual-learning network and counts only the parameter-update phase, excluding forward-pass and error computation common to both methods. The reductions in this chapter use a different denominator: complete incoming weight-row commits per sample relative to naive all-state adaptation in the same SNN and stream. They therefore establish a complementary reduction in persistent-memory traffic, not an end-to-end energy or speed advantage over these processors.

Reducing logical writes also changes the memory-technology design space. Continuous all-sample STDP couples W to a high-endurance writable memory, even though the array spends most of its lifetime retaining

a learned representation. A bounded write rate can make dense nonvolatile alternatives more plausible for the large array while retaining small, frequently updated states in high-endurance memory. This implication must be stated cautiously. Nonvolatile technologies differ in endurance, retention, write energy, variability, asymmetry, and peripheral overhead [69]; an 80% reduction in logical commits does not automatically imply an 80% lifetime increase. Device-level lifetime depends on pulse schemes, wear distribution, verification, and the selected technology.

The estimates in this section are also distinct from the FPGA results of Chapter 4. They combine simulated network activity with a published technology reference and isolate update-memory access. They exclude the control logic for confidence and selectivity, leakage, interconnect, peripheral circuits, and inference energy. Chapter 4 reports the separate implementation boundary of the realized accelerator; a full mixed-memory comparison requires a process-specific silicon design.

2.9 Discussion and Limitations

The results show that minimal plasticity is not merely a hardware heuristic. Routing states according to the reliability of their learning signal improves minority-class behaviour while reducing writes. The slow representation is modified only when the network judges its prediction to be reliable; uncertain samples are restricted to cheaper states that correct competition and assignment. The sample-relative selectivity comparison further reduces the candidate commits by concentrating them on the less-selective neurons involved in each input. It does this without a fixed convergence threshold or a persistent stop state.

Several limitations bound the conclusions. First, the controlled MNIST benchmark isolates imbalance and covariate shift but does not reproduce the temporal structure, artifacts, or inter-patient variability of a clinical biosignal. Second, confidence is empirically calibrated and may become miscalibrated under a severe or open-set shift. Third, a high selectivity value can reflect majority exposure or an erroneous assignment rather than a genuinely useful feature; comparison with the per-sample mean reduces commits but does not prove convergence. Fourth, the reported hardware saving is an access-level estimate rather than measured energy. Finally, the evaluation measures performance during individual covariate shifts and the associated update pressure; it does not measure retention on earlier tasks after sequential training and therefore does not directly test catastrophic forgetting. These limitations motivate validation on biosignal streams, explicit confidence calibration under larger shifts, sequential-shift retention experiments, and process-specific mixed-memory implementation.

This anchor interpretation also suggests a possible benefit under successive shifts: receptive fields protected during earlier adaptation may continue to provide stable class evidence while less-selective neurons respond to a later shift. The present experiments evaluate individual covariate shifts, however, and do not establish retention across sequential shifts.

Optional sparse supervision. The controller and principal results reported above are label-free. As a separate exploratory extension, a device with access to occasional expert feedback could request the true class for a small selected subset of the lowest-confidence samples and use the returned labels only in place of the corresponding pseudo-labels when updating r. In a digit-3-versus-background rotation experiment, replacing the pseudo-labels of the lowest-confidence 8% of the stream increased accuracy from 83% to 87% and minority sensitivity from 75% to 84%. The latter exceeded the upper standard-deviation range of the fully label-free all-digit rotation result. That visual cross-check is not a matched comparison, however, because the extension uses true labels and was evaluated for only one target digit; it is therefore not part of the main comparison in Section 2.7. A biomedical precedent for this optional operating model is provided in [70]: high-entropy or seizure-predicted EEG windows are sent for expert annotation, and updates occur only after enough informative samples have accumulated. On CHB-MIT, the study reports an average of 6.46 minutes of newly labelled data per day and 6.28 model updates per day, but these costs exclude the one-hour labelled patient-adaptation period required at the beginning of every experiment. Its time-window costs cannot be converted directly to the 8% sample fraction used in the MNIST extension.

A further future direction is a bounded initial calibration window. Both the assignment a_j and confidence c are functions of r, so response statistics inherited from source-domain pretraining can misroute the earliest deployment samples before r reflects the new subject or environment. During a short labelled window, true labels could be used to update r while W remains protected; the system would then transition permanently to the label-free dual-path rule. This organization has precedents in biomedical adaptation. In [71], a patient-specific feature template is initialized from a small set of normal samples, and its decision threshold is set using small normal and abnormal calibration sets before lightweight template adaptation continues. The framework in [70] similarly begins with labelled patient-specific adaptation before its selective continual-update phase. These works support calibrating compact decision state before long-term autonomous operation, but the required window length, its effect on early confidence error, and whether the benefit persists across shifts remain to be established for the present SNN.

A second preliminary diagnostic examines how many excitatory neurons are involved in each decision.

It uses the same held-out post-pretraining test set as the confidence analysis in Figure 2.2, rather than a shifted online-adaptation stream. As shown in Figure 2.8(a), the median number of neurons involved is three. The one-neuron bin contains 12.1% of the samples and achieves only 70.3% accuracy. In contrast, accuracy is above 94% for each participation count from two through six; the tail from seven through ten neurons collectively contains less than 1% of the samples and is observed to contain no errors. Panel (b) resolves the one-neuron subset into four equal-width confidence intervals. Its first three intervals achieve 52%, 41%, and 68% accuracy, respectively, whereas the highest confidence interval contains 51.4% of the subset and reaches 86% accuracy. Neuron count should therefore not be used as an independent hard gate: even decisions involving only one neuron can be reliable when confidence is high. Neuron participation instead appears complementary to confidence. A future controller could widen the abstention band when the decision involves very few neurons and return toward the base confidence bounds as more neurons participate. This relationship must first be validated across target digits, shifts, and random seeds before it is incorporated into the adaptation policy.

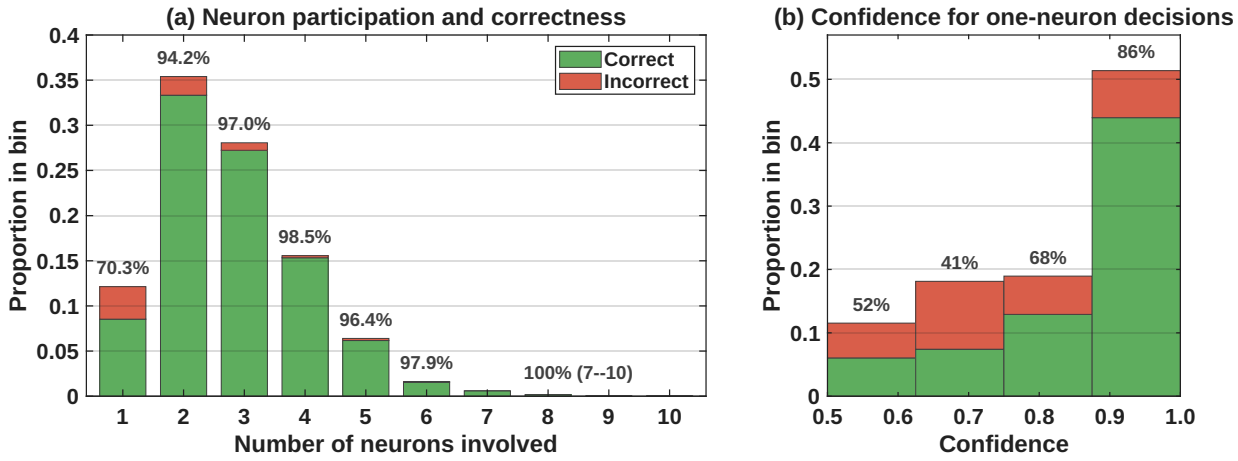


Figure 2.8: Preliminary neuron-participation diagnostic on the held-out post-pretraining test set. (a) Correct and incorrect proportions versus the number of neurons involved. Percentages above the bars give accuracy within each participation count. (b) Correct and incorrect proportions across four equal-width confidence intervals, normalized within the one-neuron subset. The one-neuron subset is less reliable overall than the higher-participation bins, but its high-confidence decisions are substantially more accurate, motivating joint rather than independent use of participation and confidence.

2.10 Chapter Summary

This chapter showed that the three persistent states of the baseline STDP-SNN should not be treated as one monolithic plastic system. A freeze ablation identified W as a slow consolidative representation and θ and r as faster tracking states. Confidence provided a network-internal reliability signal, and a confidence-

conditioned ablation derived a dual-path rule: update W on confident samples, update θ and r on non-confident samples, and abstain in the intermediate band. On the confident path, the mean selectivity of the neurons active in each sample supplied a reference that requires no selectivity hyperparameter: candidate commits from above-mean neurons were suppressed, while the less-selective active neurons remained plastic. Across controlled rare-event covariate shifts, this policy improved mean minority sensitivity relative to both no adaptation and naive all-state adaptation, and reduced weight-update pressure by about 80% during active adaptation and about 92% at steady state. Although the resulting memory-energy figures are estimates, the measured reduction in update pressure supports the design principle carried forward in the thesis: update each state only when its supervisory signal is reliable, and avoid persistent writes that add little to the current decision.

3 Class Imbalance in Spiking Neural Networks

This chapter examines how severe class imbalance affects unsupervised WTA-STDP learning and develops local, ratio-agnostic mechanisms to protect rare-class representations. It first diagnoses the failure through per-neuron selectivity and maturity, then introduces controls for first-spike competition, inhibition, assignment memory, and weight consolidation. Finally, it organizes the excitatory layer topographically so that inference can be restricted to a first-spike-anchored neighborhood.

The problem is motivated by continuous biosignal monitoring, in which clinical events such as epileptic seizures and cardiac arrhythmias occupy only a small fraction of a long recording. In an unsupervised network there is no supervised loss to reweight, and WTA competition determines which neuron learns from each input. Under a majority-dominated stream, repeated majority wins can change both the output statistics and the allocation of the finite neuron population. The experiments in this chapter study that mechanism using imbalanced MNIST as a controlled rare-event benchmark rather than clinical recordings.

The proposed mechanisms act during STDP training and use state already maintained by the network. They require no resampling, replay buffer, class-weighted loss, or estimate of the imbalance ratio. At an imbalance ratio of 1:1000, they recover minority-class sensitivity from 41% to 72%. A 9×9 inference window retains 68% sensitivity while reducing modeled per-inference energy from 6.7 to 1.8μJ.

A conference version of the core method and experiments developed in this chapter has been accepted for publication [72].

The general neuron model, input encoding, STDP rule, adaptive threshold, and Diehl–Cook architecture are introduced in Chapter 1 and are not repeated here. This chapter describes only the state variables and modifications needed for the imbalance study. Its experiments concern imbalanced training from scratch. Section 3.8.1 discusses how the mechanisms could interact with the selective adaptation of Chapter 2, but does not claim a separately validated solution for long-term imbalanced adaptation.

3.1 Prior Work on Imbalanced Learning

Class-imbalance methods are commonly divided into data-level, algorithm-level, and ensemble approaches [39, 40]. Data-level methods rebalance observations through over-sampling, under-sampling, or synthetic generation. Algorithm-level methods modify the loss, sample weight, decision threshold, or misclassification cost, while ensemble methods combine specialized learners or rebalanced partitions. In a streaming setting, ROSE

combines per-class sliding windows, imbalance-dependent Poisson resampling, random feature subspaces, drift detection, and a background ensemble [73]. These methods can be highly effective, but generally assume labeled observations and introduce stored examples, class-frequency estimates, several classifiers, or a global optimization objective.

Within SNNs, Zhou et al. modify reward-modulated STDP by applying imbalance-aware reward and punishment coefficients [74]. Their convolutional SNN associates output neurons with classes and uses the true label and network decision to select a global reinforcement signal. Experiments on Fashion-MNIST, CIFAR-10, and medical images show improved minority learning and convergence, including a Fashion-MNIST experiment in which a selected class is reduced to 1% of its original population. The work demonstrates that imbalance can be treated within an STDP update rather than only through resampling. Its correction, however, relies on supervised outcomes, class-specific output populations, and a globally routed reward.

The closest prior work is the adaptive-homeostasis method of McCarthy et al. [75]. Their two-layer, time-to-first-spike network uses causal STDP and a threshold regularizer A_{th}. In the baseline, A_{th} is fixed and every spike raises the corresponding output neuron's threshold. In the adaptive model, A_{th} is updated separately for each neuron from its spike count over a logging interval; the MNIST experiment uses an interval of 10,000 presentations so that an underrepresented class can occur before the next update. This relaxes the equal-activity assumption of ordinary homeostasis and permits different neurons to sustain different firing rates.

McCarthy et al. first demonstrate the mechanism on a two-pattern grayscale problem and then reduce one MNIST class from 100% to 5% of its original incidence, while retaining a balanced test set. At 20% incidence, their baseline becomes blind to classes 4 and 9 whereas the adaptive model retains selectivity. The benefit is concentrated near a class-dependent critical incidence: above that boundary, the adaptive model improves precision and recall in 55.9% and 57.6% of the reported cases, respectively. The authors also identify sensitivity to the initial A_{th} value and to the sparse encoding. This paper establishes that a local, unsupervised change to homeostasis can protect infrequent patterns; therefore, this thesis does not claim to be the first unsupervised or homeostasis-based imbalance treatment in an SNN. The distinction here is the explicit diagnosis of minority-to-majority capacity migration under ratios up to 1:1000, followed by a coordinated modification of competition, inhibition, assignment memory, and synaptic consolidation.

Naderi et al. separate triplet-STDP long-term weights from short-term synaptic plasticity applied during post-training [50]. Although they do not study class imbalance, their results demonstrate timescale-separated

local plasticity while identifying increased neural activity as a possible energy cost.

Other imbalance-aware SNNs mainly use supervised or global mechanisms. MeST modulates time-varying synaptic weights with a meta-neuron that combines local state and normalized postsynaptic potential while minimizing supervised output-spike-time error [76]. Wafer2Spike uses minority-specific augmentation and spatio-temporal backpropagation for long-tailed wafer maps [77]. LT-SpikingFormer combines class-balanced sampling, MixUp/CutMix, and CNN-to-SNN knowledge distillation at imbalance factors up to 100 [78]. For ECG classification, Banjo and Ghoraani combine a cascaded quantized SNN with focal loss and RR-interval features, and deploy the final inference model on a PYNQ-Z2 FPGA [79]. Federated and continual SNN methods similarly protect underrepresented labels through occurrence-probability calibration, global-model distillation, or class-balanced replay [80, 81]. Collectively, these studies show that SNN imbalance can be addressed by reward design, data transformation, global loss optimization, or memory. They do not, however, explain or control the migration of representational capacity inside an unlabeled WTA-STDP population without class-frequency knowledge or stored samples.

3.2 Problem Formulation and the V2 Network

The experiments use the `DiehlAndCook2015v2` model provided by BindsNET [45]. As in the network introduced in Chapter 1, a Poisson-encoded input layer projects through plastic feedforward synapses to adaptive excitatory neurons. The V2 designation refers to the BindsNET model variant, not to the imbalance mechanisms proposed later in this chapter. Its main architectural difference from the original Diehl–Cook presentation is that the explicit inhibitory population is replaced by direct recurrent inhibition among the excitatory neurons. An output spike therefore suppresses the other excitatory neurons without simulating a separate inhibitory neuron for each excitatory neuron. The functional role remains WTA competition.

The feedforward connections use trace-based PostPre STDP and are L_1-normalized after each sample. Adaptive thresholds provide firing-rate homeostasis, and the one-spike option permits at most one selected excitatory spike per timestep; when several neurons cross threshold together, BindsNET selects one by a multinomial draw. Every T_{label} samples, accumulated class-conditional spike rates are used to update neuron assignments. The imbalance study uses 400 excitatory neurons.

This software model is also the behavioral reference for the accelerator in Chapter 4. Chapter 4 implements its input synapses, recurrent inhibition, adaptive neurons, WTA selection, STDP traces, normalization, and firing-rate readout in fixed-point hardware. The imbalance mechanisms of Section 3.4 are evaluated algo-

rithmically here and are not implied by the V2 name.

3.2.1 Rare-event task and evaluation measures

We study imbalance in isolation using a binary one-versus-rest task. One class is the minority event and the remaining classes form a heterogeneous majority background. For N_{min} minority and N_{maj} majority presentations, the ratio is written N_{min}:$N_{\text{maj}} = 1$:IR. Performance is reported using

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP}+\text{FN}}, \qquad \text{Specificity} = \frac{\text{TN}}{\text{TN}+\text{FP}}. \tag{25}$$

These measures expose whether rare events are recovered without simply transferring errors to the majority class; overall accuracy alone would be dominated by the background population.

3.2.2 Per-neuron selectivity and maturity

Every T_{label} samples, the readout updates the per-class firing rates and assigns each neuron to the class on which it fires most. Two quantities describe the quality of that assignment. The selectivity of neuron i is

$$s_i = |p_i(c_1) - p_i(c_0)|, \qquad p_i(c) = \frac{r_i(c)}{\sum_{c'} r_i(c')}, \tag{26}$$

and its maturity is

$$m_i = \frac{r_i(c_i)}{Q_3(r(c_i))}, \tag{27}$$

where c_i is the current assignment and $Q_3(r(c_i))$ is the upper-quartile firing rate among neurons with that assignment. A neuron can therefore be selective but poorly exercised, or mature but responsive to both classes. Median splits on the two axes form four population groups; the high-selectivity, high-maturity group contains the most reliable detectors.

The implementation denotes the above- and below-median selectivity cohorts by S and U, and the mature and immature cohorts by O and Y, respectively. Thus (S,O) is selective and mature, (S,Y) is selective but immature, (U,O) is mature but unselective, and (U,Y) is neither selective nor mature. These are relative population groups recomputed at the label interval, not additional plastic states or fixed biological thresholds.

3.3 Diagnosis of Failure Under Severe Imbalance

Baseline minority sensitivity falls monotonically as IR increases. A conventional interpretation is that minority neurons simply become inactive. Per-neuron instrumentation instead identifies two coupled effects.

First, minority neurons are reassigned rather than merely silenced. Their number falls from 68 at 1:10 to 27 at 1:1000. On the selectivity–maturity plane, the share of minority neurons in the reliable group falls from 49% to 30%. Majority inputs recapture and retune neurons that had begun to specialize, leaving too little class-specific, well-developed capacity for the rare event.

Second, the transferred capacity is not converted into proportionally useful majority coverage. Majority-class redundancy is the percentage of majority-assigned neurons whose learned feedforward receptive field is a near-duplicate of at least one other majority-assigned receptive field, as determined by high pairwise receptive-field similarity. The measured redundancy rises from approximately 10% to 26% across the imbalance sweep. An increasing fraction of the majority population therefore encodes similar prototypes. Severe imbalance simultaneously removes minority specialists and spends the recovered capacity redundantly on the background class.

Taken together, the four panels support a causal interpretation of the baseline failure. Increasing imbalance gives majority inputs more opportunities to win the WTA competition. These wins progressively retune minority specialists, causing the loss of minority assignments in Fig. 3.1(b) and their movement out of the reliable group in Fig. 3.1(c). The recaptured neurons do not provide proportionally new majority features; instead, majority redundancy increases as shown in Fig. 3.1(d). The combined loss of reliable minority capacity and duplication of majority capacity produces the sensitivity decline in Fig. 3.1(a). The value of this diagnosis is not that resampling can never help; rebalancing the data would clearly change exposure. Rather, it identifies the internal allocation decisions that must be controlled when resampling and labeled class weights are unavailable.

3.4 Imbalance-Robust Plasticity Mechanisms

The four mechanisms in Fig. 3.2 act at complementary points in the failure sequence. Collectively, they reuse the per-neuron state (s_i, m_i, θ_i) and do not receive IR as a control input.

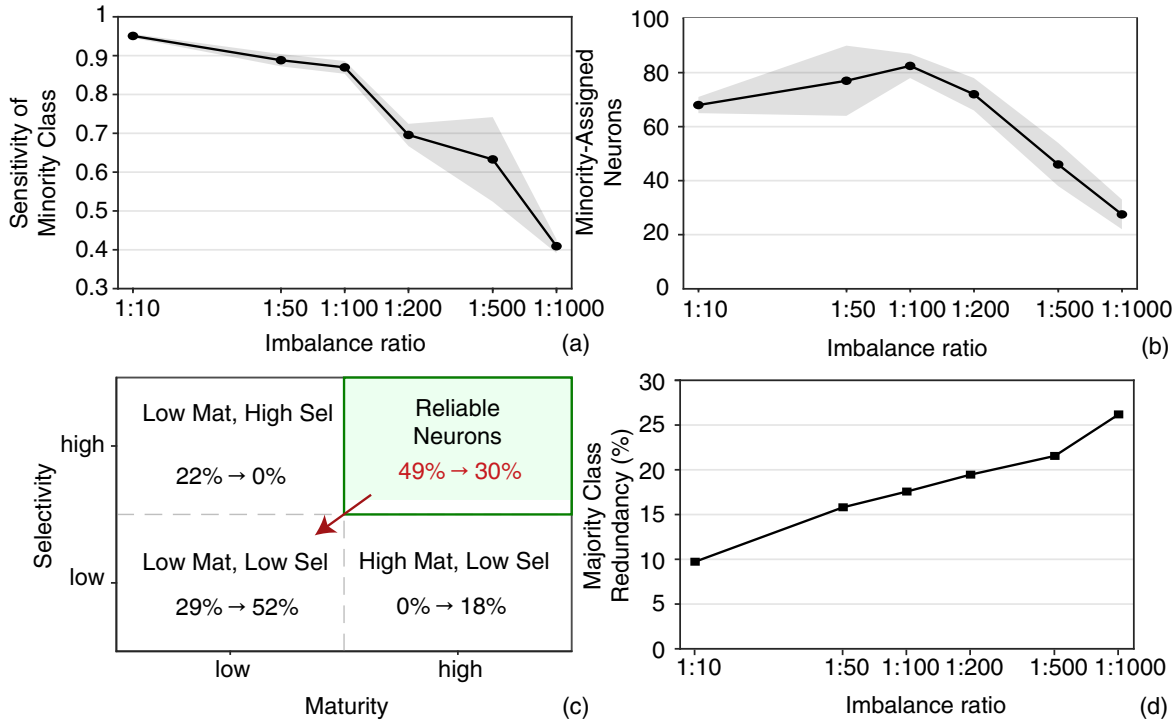


Figure 3.1: Diagnosis of the baseline as imbalance increases from 1:10 to 1:1000. (a) Minority-class sensitivity decreases across the imbalance sweep. (b) The number of minority-assigned neurons falls from 68 to 27, showing a loss of minority representational capacity. (c) Between 1:10 and 1:1000, the share of minority neurons in the high-selectivity, high-maturity reliable group falls from 49% to 30%. Over the same change, the low-maturity/high-selectivity group falls from 22% to 0%, the low-maturity/low-selectivity group rises from 29% to 52%, and the high-maturity/low-selectivity group rises from 0% to 18%. (d) The fraction of majority-assigned neurons with a near-duplicate receptive field rises from approximately 10% to 26%, indicating that the capacity lost by the minority is used increasingly to duplicate majority representations.

3.4.1 Drive-gated first-spike competition

The first output spike on a sample inhibits the remaining neurons and largely decides which receptive field learns from that input. In the baseline, a multinomial draw resolves a tie when several neurons first cross threshold at the same timestep. A poorly matched neuron can therefore win before enough evidence has accumulated; STDP then pulls its receptive field toward the input and inhibition blocks an immediate correction. The proposed rule makes only the first contested decision deterministic:

$$i^*_{\text{first}} = \arg\max_{i \in C(t)} u_i(t), \tag{28}$$

where $C(t)$ is the set of neurons crossing threshold at timestep t. Subsequent contested timesteps remain stochastic. The rule adds no persistent state and reduces to the baseline when $|C(t)| = 1$.

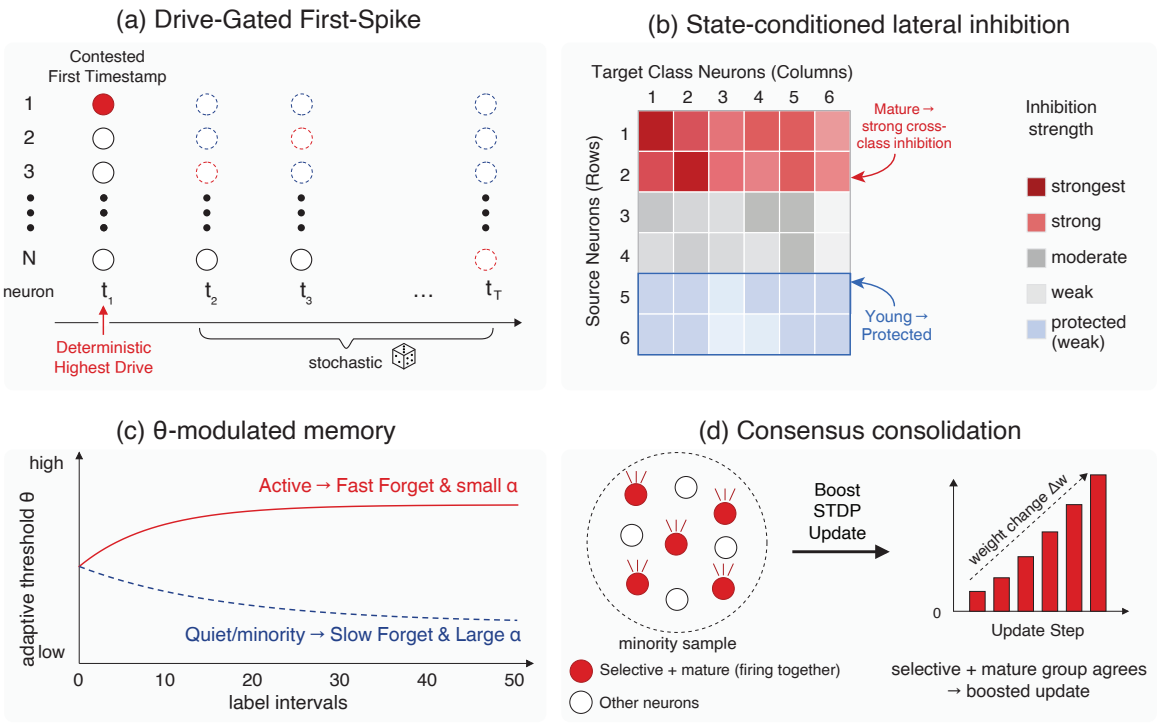


Figure 3.2: The four proposed mechanisms: drive-gated first-spike competition, state-conditioned lateral inhibition, adaptive-threshold-modulated assignment memory, and consensus-gated weight consolidation.

3.4.2 State-conditioned lateral inhibition

Uniform inhibition gives every neuron equal competitive authority, irrespective of whether its receptive field is stable and class-specific. The proposed matrix is recomputed at every label interval. For a cross-class pair, the recurrent weight is

$$W_{\text{rec}}[i,j] = -w_{\text{inh}}\sigma^{\text{src}}_{g(i)}\sigma^{\text{tgt}}_{g(j)}, \qquad i \neq j,\ c_i \neq c_j, \tag{29}$$

where $g(i)$ identifies the selectivity–maturity group of neuron i. The source gain represents how much competitive authority the presynaptic neuron has, while the target gain represents how strongly the postsynaptic neuron responds to that inhibition. Their product therefore strengthens or weakens only the relevant cross-class interaction: reliable sources can exert greater authority, while the target gain controls which competing populations are protected or suppressed.

For same-class inhibition, source trust is fixed at one for every state group. The source term therefore disappears, and the interaction is controlled entirely by the state-dependent target response listed in Table 3.1. This is why a separate full-gain equation is unnecessary.

Table 3.1: State-conditioned inhibition gains used in the implementation

Neuron group	(S,O)	(S,Y)	(U,O)	(U,Y)
Cross-class source trust	1.50	1.10	1.00	1.00
Cross-class target response	0.70	0.85	1.80	1.10
Same-class source trust	1.00	1.00	1.00	1.00
Same-class target response	1.00	0.90	1.00	0.80

Several representative cases illustrate the resulting competition. Within a class, the source state has no effect because its trust is always one. A young, unselective (U,Y) target receives gain 0.8, and a young, selective (S,Y) target receives gain 0.9, giving developing neurons modest protection from their same-class competitors; mature targets retain the baseline gain of one. Across classes, an (S,O) source acting on an (U,O) target produces the strongest gain, $1.5 \times 1.8 = 2.7$: a reliable detector strongly suppresses a mature but unselective competitor. At the opposite extreme, an (U,Y) source acting on an (S,O) target has gain $1.0 \times 0.7 = 0.7$, limiting the ability of an unreliable source to suppress a reliable target. Finally, an (S,O) source acting on an (U,Y) target has gain $1.5 \times 1.1 = 1.65$. The reliable source retains an advantage, but the reduced (U,Y) response avoids indiscriminately suppressing the large young, unselective population during majority traffic.

3.4.3 Adaptive-threshold-modulated assignment memory

The firing rates used for assignment are updated every label interval by

$$r_i(c) \leftarrow \alpha_i r_i(c) + \widetilde{r}_i(c), \tag{30}$$

where $\widetilde{r}_i(c)$ is the count from the newest interval. A single retention factor creates opposing problems. Majority neurons benefit from forgetting saturated old evidence, whereas a minority neuron may pass through many intervals without a rare-class example and can lose its assignment after a few off-class spikes. The retention factor is therefore modulated by the change in adaptive threshold:

$$\Delta\theta_i = \theta_i^{(\ell)} - \theta_i^{(\ell-1)}, \qquad \alpha_i = \frac{\alpha_0}{1 + k\Delta\theta_i}, \tag{31}$$

where ℓ indexes label intervals and $k > 0$ sets the modulation strength. A recently active neuron has a rising threshold and retains less old evidence; a quiet neuron retains its assignment history longer. Here, α_0 is the

baseline assignment-retention coefficient defined for the unmodulated readout.

In the implementation, Eq. (31) uses $\alpha_0 = 1$ and $k = 10$, and the resulting retention factor is limited to $0.5 \leq \alpha_i \leq 1.2$.

3.4.4 Consensus-gated weight consolidation

For each co-firing set $S(t)$, the consensus evidence is

$$V(t) = \sum_{i \in S(t)} s_i m_i. \tag{32}$$

The value is high only when several responding neurons are both selective and mature. For minority samples, the STDP update is multiplied by a bounded smoothstep scale. A coherent response can sharpen its receptive fields from a rare exposure, while a mixed or immature response remains near the baseline update magnitude. This mechanism is most effective at moderate imbalance, from 1:10 to 1:100; at 1:1000 the consensus condition is rarely met. It is therefore an optional component rather than a requirement of the complete method.

One implementation uses consensus knees of $V_L = 3$ and $V_H = 6$, with a maximum update scale of four. Scores at or below the lower knee leave STDP at its baseline magnitude, scores at or above the upper knee apply the fourfold scale, and values between the knees can be mapped smoothly between these two limits.

Table 3.2 maps each mechanism to the failure it addresses and distinguishes the signal-refresh time from the point at which the mechanism affects learning or competition. None adds a new per-synapse plastic state; conditioned inhibition may cache one derived group tag per neuron, while the only new history variable is the previous adaptive threshold required by the assignment-memory rule.

3.5 Topographic Organization and Window-Restricted Inference

Window-restricted inference is coupled to the preceding plasticity mechanisms rather than being an independent energy-saving stage. The state-conditioned gains establish which neurons should control competition, and the drive-gated first spike provides the spatial anchor from which the active window is selected.

Inference cost is dominated by reading the input weights of all N excitatory neurons for each input spike. Under uniform, non-topographic inhibition any neuron may win, so every column of the weight array must remain available. To localize the response without discarding the state controls, a distance factor is applied

Table 3.2: Failure-to-control mapping and state cost of the four mechanisms

Mechanism	Failure addressed	Signal, control, and timing	Stored state
Drive-gated first spike	A poorly matched neuron wins the first stochastic tie	Reads $C(t)$ and $u_i(t)$; selects the winner at the first contested timestep of a sample. Later ties are unchanged.	None
State-conditioned inhibition	Uniform authority enables specialist recapture and redundant majority allocation	Reads c_i and $g(i)$ derived from (s_i, m_i); refreshes gains each label interval and applies them whenever an output spike inhibits competitors.	Derived group tag $g(i)$; no new per-synapse state
θ-modulated memory	Rare-class evidence is overwritten between minority exposures	Reads $r_i(c)$ and $\Delta\theta_i$; updates retention α_i and the class-rate memory at each label interval.	Previous θ_i: one scalar per neuron
Consensus consolidation	A confident rare exposure provides insufficient reinforcement	Reads $S(t)$ and (s_i, m_i); optionally scales the per-sample STDP update when consensus $V(t)$ is high.	None persistent; $V(t)$ is transient

on top of the cross-class inhibition in Eq. (29). Following the self-organizing principle of Hazan et al. [82],

$$W_{\text{rec}}[i,j] = -w_{\text{inh}}\sigma^{\text{src}}_{g(i)}\sigma^{\text{tgt}}_{g(j)}\rho(d(i,j))\,\mathbf{1}[c_i \neq c_j], \tag{33}$$

where $\rho(d)$ increases competition with lattice distance. The source and target gains continue to determine competitive authority and susceptibility; $\rho(d)$ only adds the spatial organization. Neighboring neurons consequently develop related receptive fields, whereas distant regions compete.

This organization is particularly useful for the minority cohort. Minority detectors responding to related rare patterns are encouraged to occupy nearby lattice positions, so a window centered on one such detector can retain a local pool of minority-responsive neurons rather than an isolated neuron. Majority neurons with related low-level receptive-field features may also lie near this pool, but proximity does not remove the state-conditioned protection of Table 3.1. Selective, mature minority targets remain less susceptible to unreliable cross-class sources, while immature minority targets receive reduced same-class inhibition and have time to specialize.

During the search phase the full layer remains active. The first spike then anchors the responding region. If the first spiking timestep is contested, the drive-gated rule in Eq. (28) selects the neuron with the greatest membrane drive rather than a random member of the tied set. Its coordinate is therefore a more credible window center; without this rule, a random early winner could lock inference onto the wrong region. After the first spike, only an $n \times n$ neighborhood centered on its location continues to integrate, and the remaining

neurons are held at reset. If the search phase occupies fraction τ of the presentation and S_{in} input spikes are processed, the dominant synaptic-read work is

$$\text{SynOps} = S_{\text{in}} \left[\tau N + (1-\tau) n^2 \right]. \tag{34}$$

The ideal reduction approaches N/n^2, but the realized saving depends on how early the first spike occurs. A weakly driven sample searches longer and saves less energy.

3.6 Experimental Methodology

The experiments use the V2 BindsNET model defined in Section 3.2. Poisson spike trains from MNIST drive 400 adaptive excitatory neurons with recurrent inhibition. Digit 3 is the minority class and the remaining nine digits form the majority class. Digit 3 was selected because the class-wise error analysis in [44] identifies it as one of the more difficult MNIST digits and shows confusion with visually related digits, particularly 5 and 8. Because these confusable digits remain within the aggregated background class, this selection provides a challenging minority-versus-background case rather than an artificially easy separation. The controlled sweep uses IR values of 1:10, 1:50, 1:100, 1:200, 1:500, and 1:1000 without application-specific signal confounds. Unless stated otherwise, results follow 60,000 training samples. The 30,000-sample runs provide a shorter-exposure comparison, while longer runs and multiple-epoch training test whether additional exposure alone repairs the baseline failure. Performance bands summarize five random seeds.

The training experiments use the available MNIST labels at each label interval to update neuron-to-class assignments and use them again to calculate sensitivity and specificity. The feedforward weights are still learned by local STDP rather than a label-derived error. If external labels are removed after pretraining, the network must retain these assignments or obtain an unlabeled estimate with which to update them. This chapter does not evaluate such an update. Accordingly, it directly validates imbalanced pretraining; use during fully unlabeled long-term adaptation remains an intended extension rather than a separate result.

MNIST is used as a controlled diagnostic substrate rather than as a biomedical input. Long-duration implanted recordings confirm that clinically relevant events can occupy a small fraction of monitoring time [83, 84], but the ratio depends on the event definition, segmentation window, patient, and recording protocol. The sweep used here is therefore described as a controlled severe-imbalance stress range rather than a universal clinical prevalence range.

Table 3.3: Chapter-specific experimental and mechanism parameters

Parameter	Value
Excitatory population	400 neurons
Presentation duration	250 timesteps
Training length	60,000 samples unless stated otherwise
Imbalance sweep	1:10, 1:50, 1:100, 1:200, 1:500, 1:1000
Random seeds	5
State-group boundaries	Population medians of s_i and m_i
Assignment-memory gain	$k = 10$
Assignment-memory bounds	$\alpha_{\min} = 0.5$, $\alpha_{\max} = 1.2$
Consensus knees	$V_L = 3$, $V_H = 6$
Maximum update scale	$\gamma_{\max} = 4$
Selected inference window	9×9 neurons

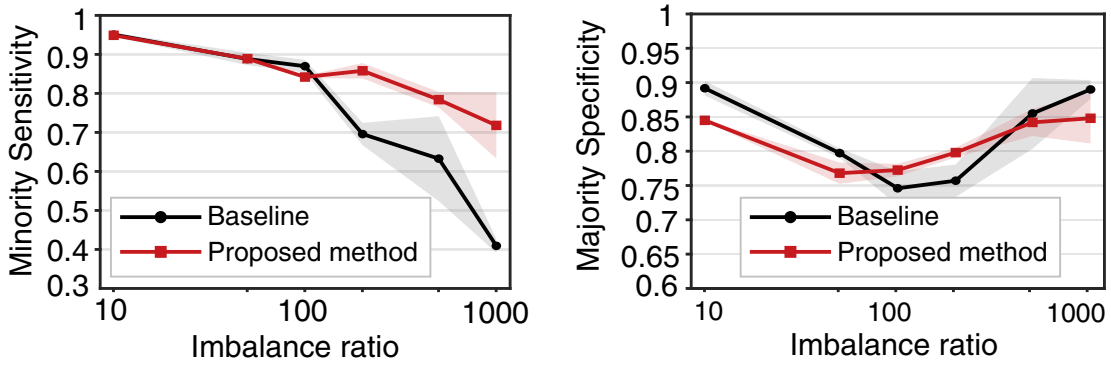


Figure 3.3: Minority-class sensitivity and majority-class specificity versus imbalance ratio. Bands show variation across five seeds.

The chapter-specific parameters are collected in Table 3.3; baseline parameters already defined in Chapter 1 are not duplicated.

3.7 Results

Figure 3.3 reports sensitivity and specificity across the imbalance sweep. Baseline minority sensitivity falls from 95% at 1:10 to 41% at 1:1000. The proposed mechanisms retain sensitivity across the sweep and reach 72% at 1:1000. Majority specificity remains close to the baseline; at the most severe ratio, the sensitivity recovery carries a four-percentage-point specificity cost. Additional training does not close the baseline gap: at 1:1000, baseline sensitivity remains 41% when training is extended through more samples or multiple epochs. The improvement is therefore not reproduced simply by extending training.

Figure 3.4 examines the internal population at 1:1000. In the baseline, minority-assigned neurons are dispersed across the four groups. With the proposed mechanisms, minority neurons concentrate more strongly

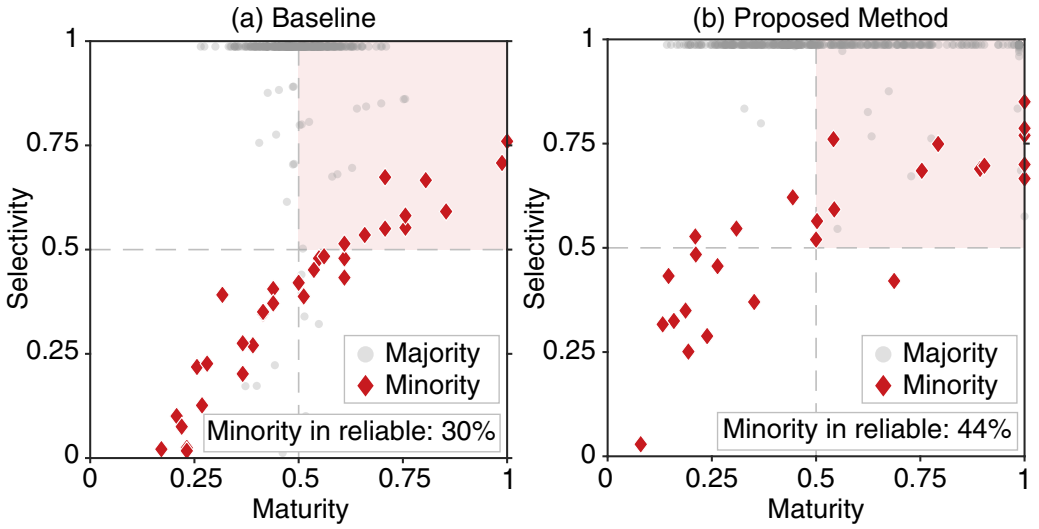


Figure 3.4: Selectivity–maturity population at 1:1000. Baseline minority neurons are dispersed across the groups; the proposed mechanisms concentrate them more strongly in the reliable group.

in the high-selectivity, high-maturity group, recovering the selective and well-developed detectors lost by the baseline. Their share in this reliable group rises from 30% to 44%, a recovery of 14 percentage points. This population-level change connects the sensitivity improvement to the failure diagnosed in Section 3.3.

Inference energy is estimated with the 45 nm, 0.9 V operation costs reported by Horowitz [41]. Because spikes are binary, a synaptic event requires a weight read followed by an accumulation. Reading one 8-bit weight from the approximately 306 kB input-to-excitatory array costs about 7.1 pJ, compared with approximately 0.05 pJ for the accumulation. When the full layer is active, every input spike therefore reads all N input-to-excitatory weights. In the worst case, $S_{\text{in}} = N_{\text{in}}T$, the resulting estimate is approximately 789 μJ per sample, of which approximately 99% is due to weight reads. Under the standard Diehl–Cook rate coding [44]—pixel-proportional Poisson rates up to 63.75 Hz over 250 timesteps, producing approximately 1.6 thousand input spikes per sample—the full-layer estimate is 6.7 μJ per sample, with approximately 97% attributed to weight reads. The four plasticity mechanisms add less than 0.1% to this inference estimate, and the distance-graded inhibition is evaluated from the stored group and class tags on each output spike at negligible modeled cost. The window is therefore the main energy lever because it reduces the input-weight fan-out from N neurons to n^2 after the first spike.

Figure 3.5 sweeps the topographic window. A 9×9 window is the smallest operating point close to the performance plateau at the most severe ratio. At 1:1000, sensitivity decreases from 72% to 68%, while modeled energy falls from 6.7 to 1.8 μJ, a 3.7$\times$ reduction. The four-percentage-point sensitivity loss is

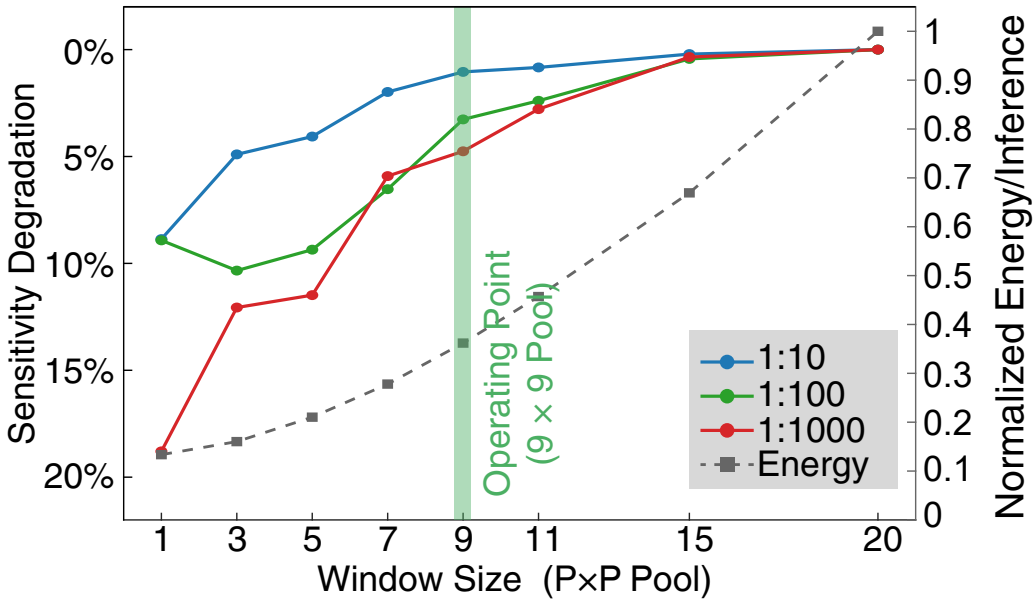


Figure 3.5: Minority-sensitivity degradation and normalized modeled inference energy versus topographic window size at imbalance ratios 1:10, 1:100, and 1:1000. The selected 9×9 window gives $3.7\times$ lower energy and approximately 6% relative sensitivity degradation at 1:1000.

Table 3.4: Operating points at 1:1000; detection values are simulation results and energy values are analytical estimates

Configuration	Sensitivity	Specificity	Energy	Reduction
Baseline, full layer	41%	89%	6.7 μJ	—
Plasticity mechanisms, full layer	**72%**	85%	6.7 μJ	—
Plasticity mechanisms + 9×9 window	**68%**	84%	**1.8 μJ**	3.7×

approximately 6% relative to the 72% full-layer value. Smaller windows save more energy but pass the knee of the sensitivity curve; larger windows recover little additional performance for the extra memory accesses. At 1:1000, the full-layer plasticity mechanisms improve minority sensitivity by 31 percentage points over the baseline. The 9×9 window retains a 27-point improvement and detects approximately $1.7\times$ as many minority events as the baseline, while using $1.8/6.7 \approx 27\%$ of the modeled full-layer inference energy. Its specificity is 84%, one percentage point below the full-layer proposed method and five points below the baseline.

3.8 Discussion

The results characterize imbalance in unsupervised WTA-STDP as a population-allocation problem. Rare-class sensitivity is lost because specialists are reassigned and the released capacity becomes redundant majority representation. The four mechanisms act at complementary points: first-spike gating improves the

initial learning decision, conditioned inhibition changes competitive authority, adaptive memory protects assignments across long gaps between minority samples, and consensus scaling strengthens coherent rare updates. The recovery of reliable minority detectors supports this interpretation.

The experiments are intentionally a controlled mechanism study: digit 3 versus the remaining MNIST classes, using one 400-neuron rate-coded configuration. This setting isolates how imbalance changes internal allocation and supports the conclusions within the tested range; evaluation on biomedical signals, other encoders or network sizes, and a native multiclass output remains follow-on validation. Likewise, ratio-agnostic means that no control law receives IR or a class-frequency estimate and that one parameter set is held across the sweep, not that the parameters are universal. The online-adaptation and hardware scope are discussed next.

3.8.1 Implications for online adaptation

An imbalanced deployment stream gives majority samples more opportunities to modify synaptic weights. The confidence and selectivity gates of Chapter 2 reduce the number of structural commits and should therefore slow majority-driven drift. This makes the present mechanisms adaptation-ready at the interface level: their state can be retained and their updates can be routed by the Chapter 2 controller. It is not yet evidence of a complete long-term imbalance solution. The combined behavior under covariate shift, imperfect pseudo-labels, and a continuing severe imbalance remains to be evaluated.

3.8.2 Hardware implications

The added controls require primarily per-neuron or periodic arithmetic rather than per-synapse state. First-spike selection performs one reduction over the contested set. Inhibition gains can be reconstructed from compact group and assignment tags. Assignment memory adds a threshold difference and bounded scale per neuron at a label interval, while consensus scaling adds a weighted reduction per sample. These costs are small compared with repeatedly reading the feedforward weight array. Nevertheless, the 6.7 and 1.8 μJ values are technology-normalized analytical estimates, not measurements from the FPGA system in Chapter 4.

3.8.3 State-aware early termination as future work

The current windowing mechanism reduces spatial work after the first spike but continues through the remaining timesteps. A possible extension is to let the state of the first-spiking neuron control temporal termi-

nation as well. A selective, mature (S, O) winner projects strong inhibition and can establish dominance in its local window quickly; a young or unselective winner should be given more time for competing neurons to accumulate evidence. This suggests a state-dependent observation period.

Historical state alone is not a safe stopping condition. A mature majority neuron can still fire first on an unfamiliar minority sample. A complete controller should require (i) an (S, O) first winner, (ii) a minimum observation time $t_{\min}$, and (iii) a sample-level confidence or winner-margin condition sustained for L timesteps. Only their conjunction would terminate the presentation; all other cases would continue normally. This is a design hypothesis rather than a result of this chapter. Its thresholds and its sensitivity–energy tradeoff require separate simulation before it can be incorporated into the architecture.

3.9 Chapter Summary

Severe imbalance damages an unsupervised WTA-STDP network by reassigning minority specialists and increasing redundant majority representation. Four local mechanisms protect the competitive decision, inhibitory authority, assignment memory, and weight update magnitude. At 1:1000, they raise minority sensitivity from 41% to 72% while maintaining 85% specificity. A first-spike-anchored 9×9 window retains 68% sensitivity and reduces modeled inference energy by $3.7\times$. These results establish a ratio-agnostic method for severe imbalanced training and spatially conditional inference. Validation on long-duration biomedical streams and during prolonged unsupervised adaptation remains future work.

4 FPGA Hardware Realization

4.1 Overview

The preceding chapters developed the algorithm: the Diehl–Cook spiking network trained unsupervised with STDP, a confidence-gated rule for selective online adaptation (Chapter 2), and mechanisms for class imbalance (Chapter 3). This chapter takes that algorithm to hardware. Deploying an adaptive spiking network on an edge or implantable device requires that both inference and learning run on-chip, within a tight energy budget and without reliance on a host, and the design presented here meets those requirements: Poisson-driven inference and unsupervised STDP learning are carried out entirely in the programmable logic of a Zynq UltraScale+ device, while the host only streams encoded inputs and collects per-sample decisions. Among the reported FPGA STDP accelerators it is one of the few to perform unsupervised learning and inference fully on-chip with power and energy measured on hardware rather than estimated.

Section 4.2 surveys prior FPGA STDP-SNN accelerators; Section 4.3 treats the fixed-point quantization the learning rule forces; Section 4.4 develops the architecture and the reasoning behind it; Section 4.5 reports the measured accuracy, resource, power, and energy figures and compares them with prior work; and Section 4.6 discusses the hardware implications of the adaptation and imbalance mechanisms and the design's limitations. Section 4.7 closes the chapter by summarizing the implementation and its measured trade-offs.

4.2 Related Hardware Work

The design space for on-chip STDP accelerators is set less by the learning rule than by how a design stores and updates its state. A competitive network needs a membrane potential, an adaptive threshold, and one or two synaptic traces per neuron, together with a large weight matrix; holding these is cheap in arithmetic but costly in memory and area. Prior FPGA work is therefore best read as different trades among processing-element count, memory traffic, and numerical precision, and the design of this chapter occupies one particular corner of that space.

Several designs implement genuine on-chip unsupervised STDP, as this work does, and contain its cost by folding the network onto a small pool of processing elements. Wu et al. [85] compute the STDP exponential with an iterative shift-and-add (CORDIC) scheme rather than a lookup table, and time-multiplex a 784×100 network onto a 14×10 array. Li et al. [64] add a useful observation, that event-driven updating is not always cheapest when spikes are dense, and switch each neuron between clock- and event-driven modes while

sharing one datapath between inference and learning. Guo et al. [86] bank the weights for parallel readout and clock-gate the learning datapath during inference, both choices this work shares. The cost of time-multiplexing in all three is memory traffic: because the leaky membrane of every neuron must be updated on every timestep, folding the network onto a few processing elements turns per-neuron state into a stream of read-modify-write accesses, and the bottleneck moves from arithmetic to state memory and per-image latency, about 6.1 and 16 ms for a learning image in [85] and [64]. A fully parallel datapath, which holds each neuron's state in its own register and updates it in place, avoids that traffic. Wu et al. meet a second limit this work does not: at 8-bit weights the smallest additive STDP step is not representable, so their rule must be made weight-dependent to scale the increment, whereas the 16-bit weights of Section 4.3 keep the additive depression step exact.

The closest architectural comparison is the design in [87], which also maps a 784-input, 400-neuron network fully in parallel and reads the corresponding weights from banked BRAM. It reaches a low inference latency of 215 μs, but training is performed offline in software and the 5-bit weights are frozen in the FPGA. For the additive STDP coefficients used in this thesis, that precision cannot retain the smallest depression updates and is therefore not an option for the on-chip learning datapath. Wider weights chiefly increase the weight-memory demand; they do not, by themselves, explain the LUT difference. The architecture and resource comparison in Section 4.5 separates the cost of weight storage from the logic required for plasticity, adaptive state, readout, and control.

A separate group reaches higher MNIST accuracy by admitting a supervisory signal, and these are fairly the most accurate designs in the field. He et al. [88] fix a random hidden layer and train only a linear output with reward-modulated STDP, reaching 93% with very little learning hardware; the design in [66] reaches 95.1%, though that figure comes from a supervised reward-modulated stage layered on top of its unsupervised clustering, and its purely unsupervised operation sits close to the range reported here; Vallejo-Mancero et al. [89] drive the plasticity with label-selected teacher spikes. Each therefore depends on labels or a reward during learning, and in the first case only the output layer is plastic at all, whereas the rule kept here is label-free, so the accuracy reported in Section 4.5 is obtained with unsupervised synaptic learning. The platform in [89] also shows a different trade: its programmability is a real strength, but storing several memory words per synapse consumes about 60% of the device's block RAM for a 364-neuron application, a cost that a fixed-function accelerator avoids.

An earlier study by Wang et al. [90] is candid about the trade between parallelism and energy, and it also

exposes two costs the present design sidesteps: it performs the synaptic operation with multipliers, although a spike merely gates a weight into an adder, and it stores potentiation and depression traces per synapse, which dominate its learning memory. Keeping traces per neuron and per input, rather than per synapse, keeps that state small.

Taken together, prior work leaves one corner of the space open: a fully parallel datapath, whose in-place per-neuron state avoids the memory traffic of time-multiplexing, paired with weights wide enough to represent the additive STDP step and a label-free rule, so that unsupervised learning and inference run together on-chip. That is the design of this chapter. Its cost is area, the recurring price of parallelism, quantified in Section 4.5; a further difference is that the accuracy, latency, and energy reported there are measured on the board, whereas most of the figures above are tool estimates.

4.3 Quantization

The number formats are fixed before the datapath, because the learning rule constrains them and the architecture then exploits those constraints. The accelerator uses scaled-integer (fixed-point) arithmetic throughout, and the weight precision is not a free parameter; it is constrained by the STDP learning rates. The rule uses a potentiation rate $\eta_{\text{post}} = 10^{-2}$ and a depression rate $\eta_{\text{pre}} = 10^{-4}$, and the stored weights use 16 fractional bits. In that format the potentiation coefficient is 655 LSB and the depression coefficient is 7 LSB. The implementation rounds both the stored coefficient and the fixed-point product. More generally, with F fractional bits the rounded full-trace depression coefficient is $\text{round}(10^{-4}2^F)$: it is 2 LSB at $F = 14$, 1 LSB at $F = 13$, and zero at $F = 12$. Thus full-trace depression survives at $F = 13$ and disappears at $F = 12$ under the implemented rounding convention. Under truncation it would instead survive at $F = 14$ and disappear at $F = 13$; under either convention, the 16-bit resolution is not the minimum bitwidth to represent depression learning step. The reason for retaining 16 fractional bits is graded-update headroom rather than binary representability. With the implemented 7-LSB coefficient, rounded depression remains non-zero for post-synaptic traces above approximately 0.07; the corresponding threshold rises to approximately 0.25 at $F = 14$ and 0.5 at $F = 13$. This sparsity is exploited by the datapath in Section 4.4.

To separate inference precision from learning precision, the trained $N = 400$ checkpoint is run in inference only after its weights are requantized to 1, 2, 3, 4, 6, 8, 10, 12, and 16 fractional bits. Figure 4.1 shows that inference accuracy has effectively saturated by four fractional bits. The additional precision is therefore retained for the trace-scaled STDP update rather than for classification alone.

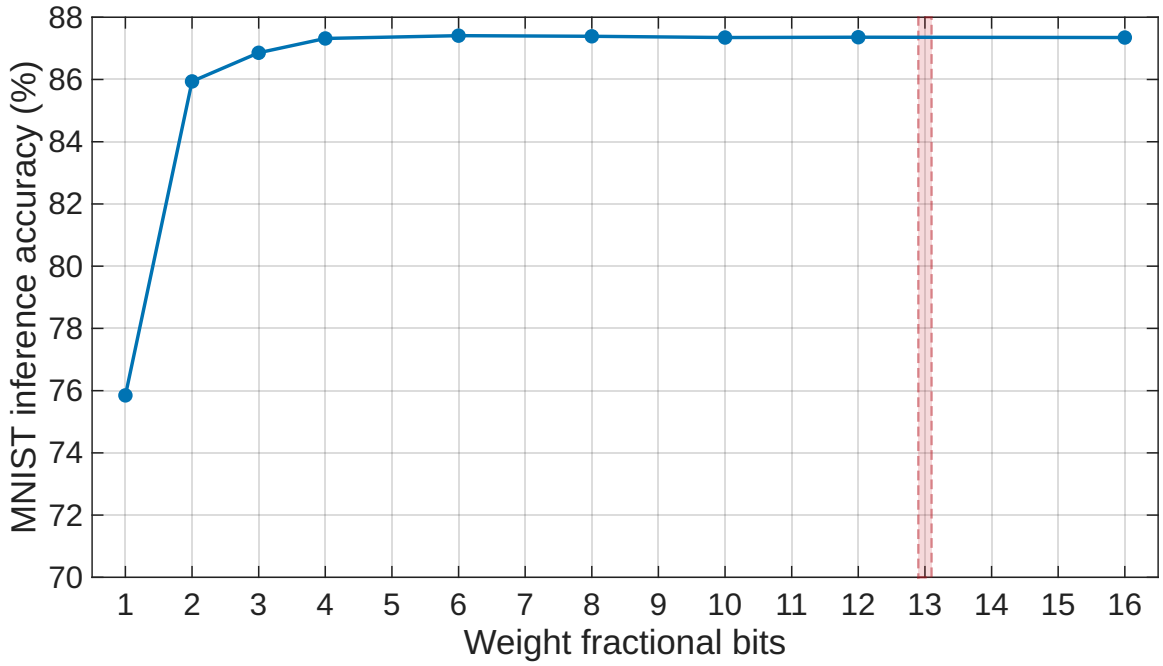


Figure 4.1: Measured inference accuracy of the reference model after requantizing the trained $N = 400$ checkpoint. The translucent marker at $F = 13$ is an analytical reference: it is the lowest precision at which the rounded full-trace depression coefficient remains non-zero, not the result of a learning-precision sweep.

The membrane potential v and the adaptive threshold θ each use a signed 32-bit fixed-point word, one sign bit with 15 integer and 16 fractional bits. The integer part is sized by the values θ reaches as the original network converges during training, which are the largest magnitudes either quantity takes. The fractional part is sized by the threshold leak, and 16 bits are the fewest that keep the once-per-sample decay of Section 4.4 representable; a coarser fraction would round that decay away. The synaptic traces are $Q0.16$ fractions in $[0,1)$, the per-neuron spike counters are 6 bits (bounded by $\lfloor T/(1+t_{\text{refrac}})\rfloor \approx 41$), and the readout rates are 16-bit integer accumulators, with the proportions kept fractional. Table 4.1 summarizes the widths. The accelerator is an optimized fixed-point realization and is not bit-identical to the full-precision reference model by design; each substitution trades an imperceptible numerical difference for a large hardware saving, and the design is validated by matching the reference model's predictions and accuracy (Section 4.5), not its bits.

4.4 Architecture

The accelerator realizes the v2 variant of the Diehl–Cook network (introduced in Chapter 3): $M = 784$ Poisson-encoded inputs drive a single population of N excitatory neurons with adaptive thresholds, and the

Table 4.1: Fixed-point widths. The 16-bit weight fraction retains headroom for graded depression; the membrane and threshold fractions are set by the once-per-sample threshold leak, and the remaining widths are bounded structurally.

State	Bits	Format	Note
Weights W	16	$Q0.16$	$\eta_{\text{pre}} = 7$ LSB; graded LTD to $x_{\text{post}} \approx 0.07$
Membrane v	32	signed $Q15.16$	membrane voltage range
Threshold θ	32	signed $Q15.16$	integer range from trained θ; fraction from the θ leak
Traces	16	$Q0.16$	values in $[0,1)$
Spike counts	6	integer	$\leq \lfloor T/(1+t_{\text{refrac}}) \rfloor$
Rates	16	integer	readout accumulator
Proportions	18	$Q2.16$	readout
Assignment	4	integer	$\lceil \log_2(C+1) \rceil$

separate inhibitory layer of the original model is replaced by a fixed recurrent connection through which any neuron that fires inhibits all of the others. STDP acts on the input weights and a labelled spike-rate readout produces the classification. The network is mapped to one datapath that performs both inference and on-chip learning, selected at run time by a learn-enable signal. Unless stated otherwise the design point is $N = 400$ neurons, $C = 10$ classes, and $T = 250$ timesteps per sample.

Two properties of the algorithm shape the design. The first is that a fully parallel mapping is costly. Each of the $M \times N$ synapses becomes a weight register whose value is gated by the presynaptic spike and accumulated into its neuron's current, and every neuron needs a wide adder to sum its M inputs; for a few hundred neurons this is already hundreds of thousands of synaptic elements and N large adder trees, exceeding a mid-range device. The second is that the computation is sparse and event-driven: the input is Poisson-coded, so only a few inputs are active each timestep, the neurons fire under a one-spike rule, and the weight update is dominated by a single winning neuron. Underlying both is that the weight matrix has far more entries than all the other state combined, of order $M \cdot N$ against $O(M+N)$, so it dominates both on-chip storage and the per-timestep data transfer; how the weights are held and moved is therefore the decision that shapes the datapath. The first property justifies storing the weights in an on-chip memory while keeping the neurons parallel, and the second an event-driven control that touches only active state. The subsections below place the accelerator on the board and then develop the datapath block by block.

4.4.1 System Integration

The accelerator is deployed on a Xilinx Zynq UltraScale+ MPSoC (ZCU102, `xczu9eg`), a 16 nm FinFET evaluation platform aimed at AI/ML workloads. It pairs a hardened ARM processing system with a large,

power-efficient programmable-logic fabric that holds the full weight matrix and the parallel datapath with room to spare (Section 4.5), and it can report the accelerator's power directly on the board rather than leaving it to estimation. As shown in Figure 4.2, the dataset is held on an SD card, and the processing system (PS, a quad-core ARM Cortex-A53) reads each sample from it into DDR memory, Poisson-encodes the image into a $T \times M$ spike raster, and streams the raster, one M-bit row per timestep, to the programmable logic (PL) over an AXI-DMA channel; at the end of a sample the PL returns a compact decision word (the predicted class and a confidence value). The PS side is purely data delivery and result collection, and the accelerator logic itself is independent of the host.

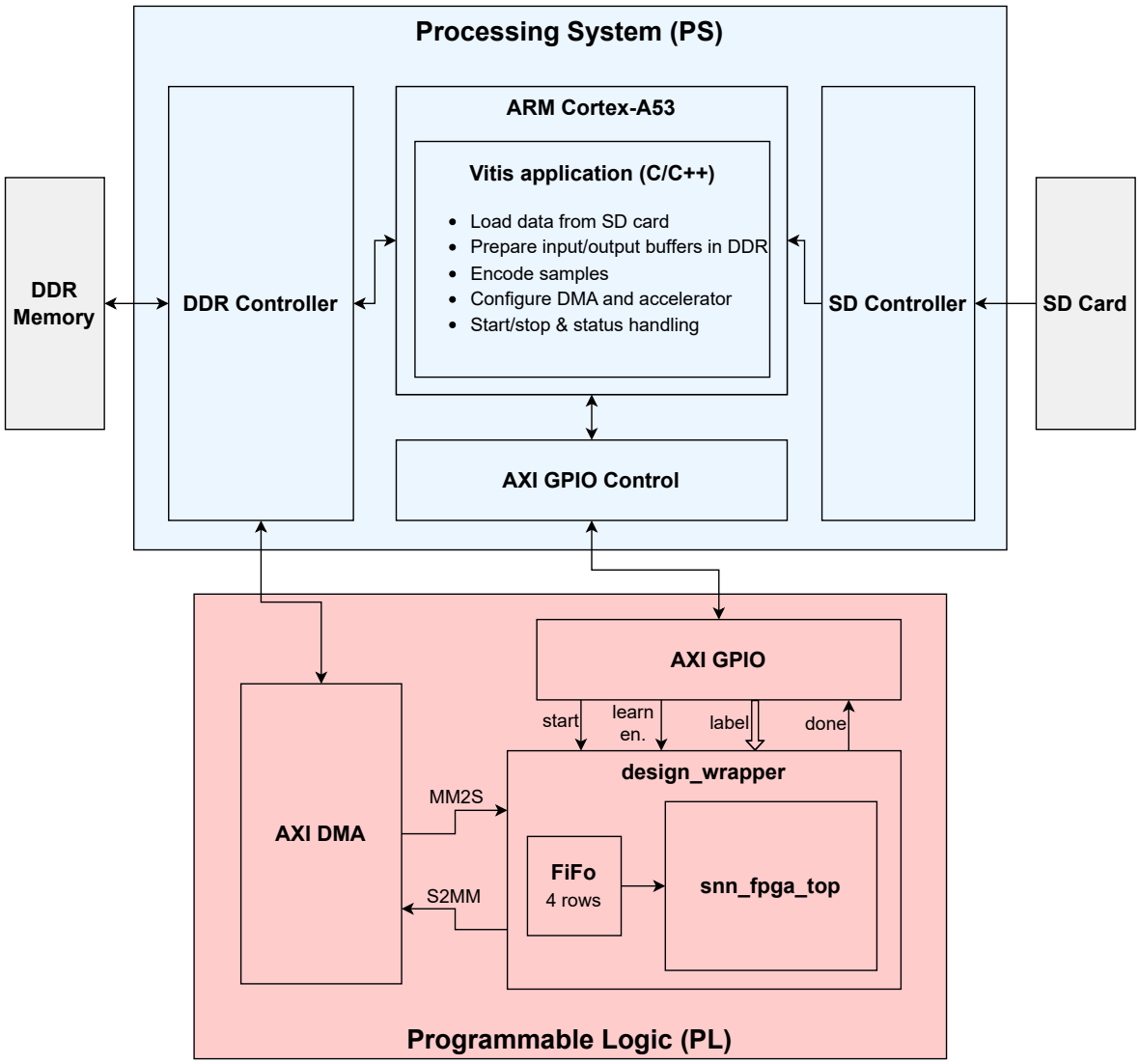


Figure 4.2: System-level integration on the ZCU102. The processing system (PS) reads samples from an SD card into DDR, Poisson-encodes each image, and streams the spike rasters to the programmable logic (PL) over AXI-DMA; the accelerator (`snn_fpga_top`) returns a per-sample decision.

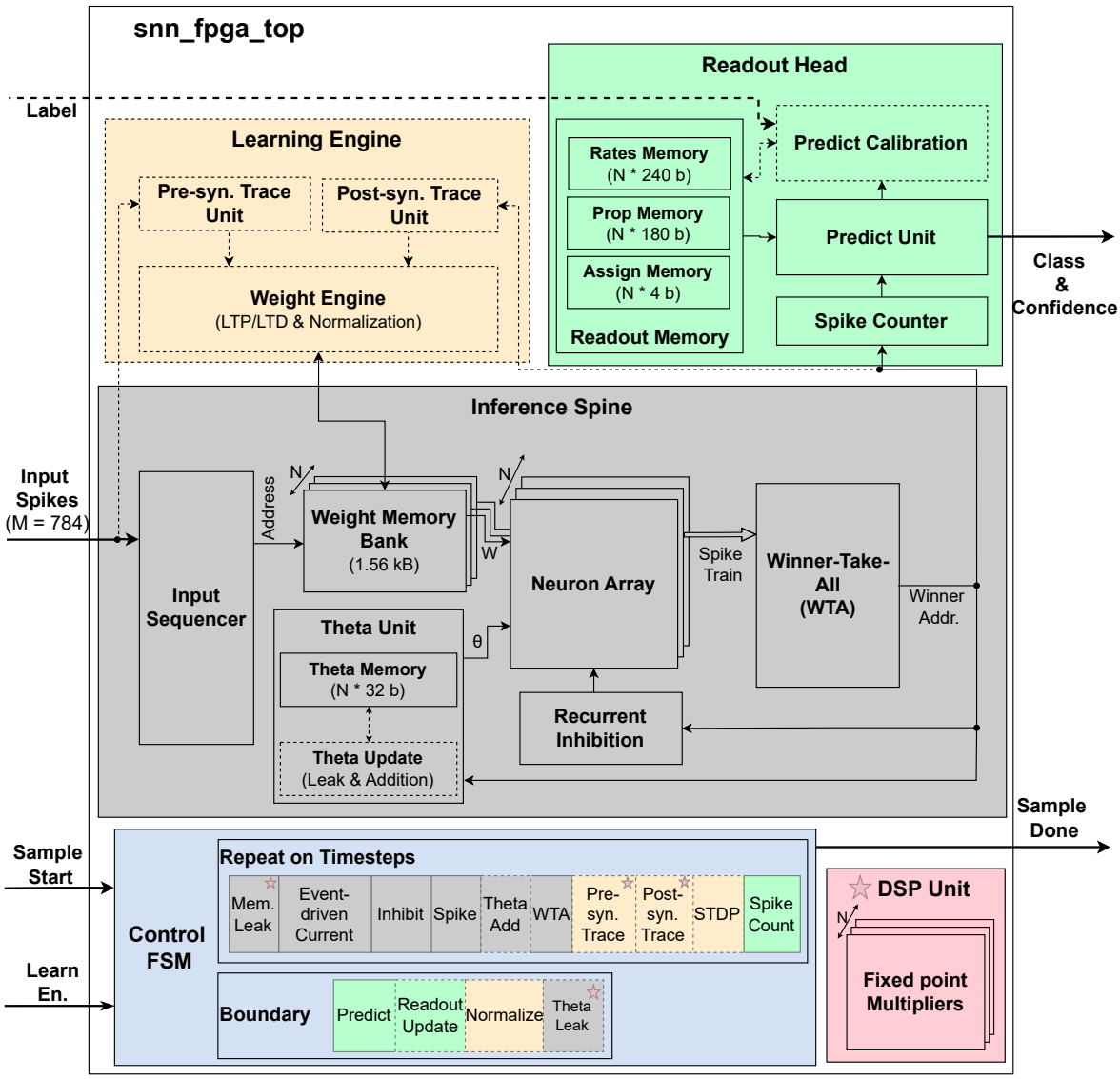


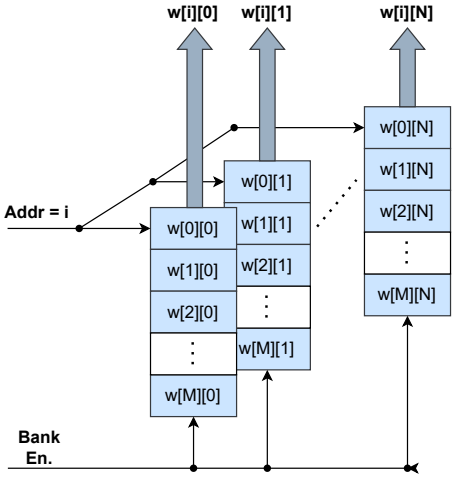
Figure 4.3: Block diagram of the accelerator (`snn_fpga_top`): the inference spine (banked weight memory, neuron array, winner-take-all, recurrent inhibition, adaptive threshold), the learning engine (pre-/post-synaptic trace units and the STDP weight-update engine with normalization), the readout head (spike counters, rate/proportion/assignment tables, prediction and confidence), and the control FSM. Dashed outlines mark the blocks gated by the learn-enable signal. The weight-memory block is labelled per bank (1.56 kB, one neuron's 784 weights at 16 bits); the full matrix at $N = 400$ is about 627 kB.

Figure 4.3 shows the accelerator, `snn_fpga_top`. It is built from three functional groups sequenced by a control finite-state machine (FSM): an inference spine that carries the forward path, a learning engine that trains the weights, and a readout head that both produces the labelled decision and maintains the rate, proportion, and assignment memories it reads from. A learn-enable signal switches the learning on or off, and the dashed outlines in Figure 4.3 mark the blocks it gates: the learning engine, the adaptive-threshold

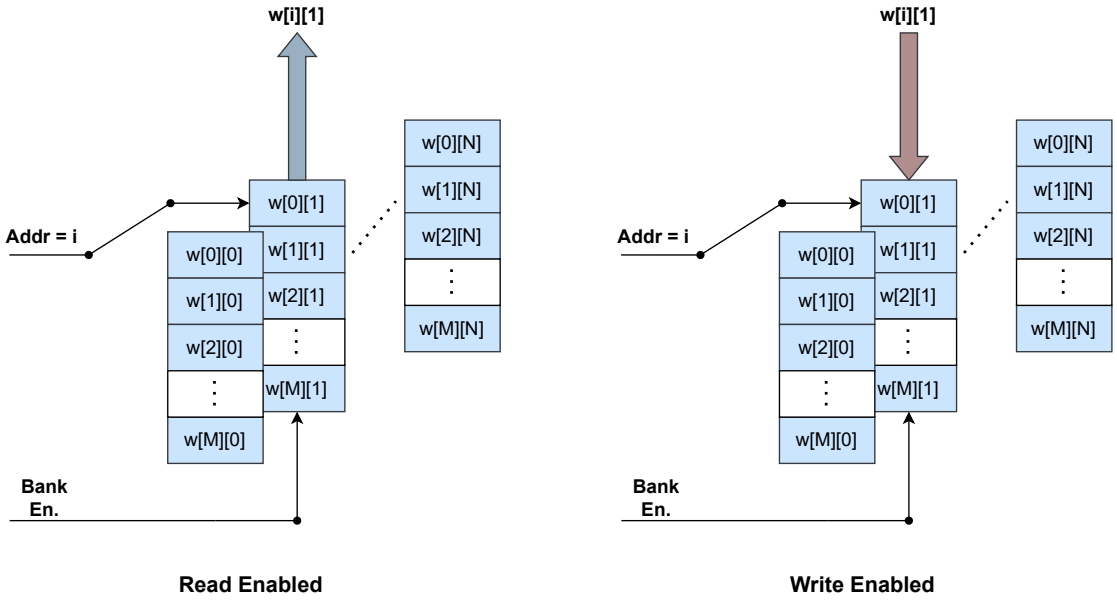
update in the lanes, and the readout-memory update. The building blocks of each group, and the decisions behind them, are described in the subsections below.

4.4.2 Weight Memory and Banking

The weight matrix is the largest of the accelerator's memories. The weight matrix is split into N banks similar to [86], one per excitatory neuron (N being the number of neurons), so that bank j holds the weight column of neuron j, a column of M entries addressed by the input index ($M = 784$, the number of input pixels). This banking is chosen for the access that runs most often. On every timestep each active input must present its weights to all N neurons at the same time, and with one bank per neuron this is a single broadcast read across all the banks. The other way the memory is used, the by-column reads and writes of the STDP update and the normalization, walks one bank at a time and only for the neurons that spiked, which happens far less often. Banking by input instead would speed up those occasional by-column operations, but it would turn the frequent broadcast read into a sequential sweep of N banks. Because that read happens on every timestep while the by-column work happens only on spikes and at the sample boundary, multiplying the dominant access by N far outweighs any saving on the rare one (up to $46\times$ more memory cycles per sample). Figure 4.4 illustrates the two access patterns.



(a) Broadcast row read, used by the per-timestep currents: one input to all N neurons in parallel.



(b) By-column read-modify-write, used by the STDP update and the normalization: one neuron's column at a time.

Figure 4.4: The two access patterns of the banked weight memory.

4.4.3 Inference Spine

The forward path runs on the neuron array, which holds N neuron units, one per excitatory neuron. Each unit caches its own membrane potential and refractory counter, so the per-neuron state is updated in place rather than streamed through a shared memory as in the time-multiplexed designs of [85, 64]. The array shares a

pool of adders and comparators, used almost entirely here, and the DSP multipliers, which are also called on by other blocks. On every timestep the control FSM drives the array through a short fixed sequence, drawn as the per-step phases in Figure 4.3.

The membrane potential leaks first. In one cycle every membrane is scaled by the same decay constant and offset back toward its resting value, as a single folded multiply-add across the array. The event-driven current follows, and only the active inputs are read, each adding its weight row into the membranes in one cycle, so the step lasts as many cycles as there are active inputs, on average about 15 for the Poisson-encoded MNIST used here. A one-cycle inhibition step comes next, subtracting a fixed value from every membrane whenever a neuron spiked and won on the previous timestep, a value that is both updated and applied in a single cycle. Following the baseline network the same inhibitory value is used for all neurons, although the feasibility of a per-class heterogeneous inhibition examined in Chapter 3 could be applied in the same one-cycle style easily.

Each membrane is then compared with its adaptive threshold θ in one cycle; a crossing neuron resets to the reset voltage and enters its refractory period, and when learning is enabled a fixed step is added to its θ as the homeostatic threshold increase. The neurons that cross θ compete under the one-spike rule so that only one fires per timestep. BindsNET draws this winner by a multinomial choice among the neurons above threshold, but a fixed lowest-index winner is enough here, which reduces the step to a combinational lowest-index-one finder in a single cycle. Starting that search from a rotating index rather than from zero would keep low-index neurons from being favoured over time, a refinement not needed for the present results.

These steps repeat on every timestep. One more, active only during learning, runs once per sample at the boundary and is a further simplification of the BindsNET model, where θ decays by a minute amount on every timestep. Because that per-step constant is so close to one, representing it faithfully would take roughly 24 fractional bits; applying the decay once per sample instead, as the constant raised to the number of timesteps, enlarges it enough to represent in the 16 fractional bits the design already uses, at a negligible difference across the sample.

4.4.4 Learning Engine

When learning is enabled, three operations join the per-timestep loop and one more runs at the sample boundary, and all of them build on the STDP trace. As introduced in Section 1.4, trace-based STDP keeps one trace variable for each input and each excitatory neuron; the trace is set to its maximum value of one

when its neuron spikes and otherwise decays toward zero. Setting a trace on a spike is a single write, but the decay has to be applied to every non-zero trace on every timestep. The traces are kept per input and per neuron rather than per synapse; a per-synapse trace, as in [90], would grow as $M \cdot N$, whereas the per-input and per-neuron form is only $O(M+N)$, a large reduction in learning state.

The M input traces are decayed on the DSP multipliers. Because those multipliers are sized to the N output neurons, decaying the input traces takes $\lceil M/N \rceil$ passes: a single cycle when $N \geq M$, and two cycles at $N = 400$. The N neuron traces are handled the same way, except that only the winning neuron of the timestep is set to one while the rest decay. Because the network fires sparsely, about 12 spikes and 2.6 active neurons per sample (Chapters 2 and 3), only a few neuron traces are non-zero at any moment and the zero ones are skipped. A trace-scan, similar to [64], reads the traces in parallel and emits the addresses of the non-zero ones one per cycle, and that address stream drives the weight update.

The weight change has two parts, potentiation (LTP) and depression (LTD), and the one-spike rule keeps both cheap. Potentiation occurs only when a neuron won the timestep, and it touches that winner's weight column alone. Walking the non-zero input traces from the trace-scan, each addressed weight is read, updated, clamped to $[w_{\min}, w_{\max}]$, and written back. When the input spiked on this timestep its trace is exactly one, so the change is the constant $\eta_{\text{post}} - \eta_{\text{pre}}$ and no multiplication is needed; when the input trace is non-zero but smaller, the change is η_{post} times that trace, a single multiply. The $-\eta_{\text{pre}}$ term folds in the winner's own depression, whose post-trace is one. Depression of the other neurons is smaller still: the trace-scan over the neuron traces gives the few non-winner neurons with a non-zero post-trace, and for each of them one multiply forms η_{pre} times its post-trace, which is then subtracted from the weight of every input that spiked this timestep, again with clamping. A single multiplier is shared across both walks. Because the post-trace is significant for at most about one neuron, depression is close to noise and the winner-column potentiation dominates, so over a sample learning adds about 4,000 cycles, comparable to inference.

At the sample boundary the input weights are renormalized so that each column sums to a fixed L_1 norm of 78.4, which is $M/10$. Only the columns of the neurons that spiked are normalized, and those neurons are already known from the spike counts kept for the readout; an unspiked column is left unchanged. For each such column, on average 2.6 per sample, the engine reads its M weights and accumulates their sum S, forms the scale $78.4/S$, and then multiplies every weight by that scale and writes it back, which needs the column to be held briefly in on-chip registers. The division is done by a sequential radix-2 (shift-and-subtract) divider that produces one quotient bit per cycle; it was chosen over a DSP-based reciprocal because

the divisor is wide and the normalization runs only at the boundary, where spare cycles are plentiful, so the slow divider costs nothing in throughput. Reading and rewriting one column takes about $2M$ cycles, and the whole boundary normalization takes on the order of 5,000 cycles per sample.

4.4.5 Readout Head

The readout turns the spike activity of a sample into a class decision, and it also keeps the running statistics that on-chip learning needs. A spike counter records, for every neuron, how many times it spiked during the sample together with the addresses of the neurons that spiked, and these counts are read both by the boundary classification and by the weight normalization. Alongside the counter, an auxiliary computation tracks a running confidence in the decision as the sample proceeds. The confidence is reported with the classification, but its main purpose is to be the signal on which the confidence-based adaptation of Chapter 2 is built, so the readout produces it here even though the present accelerator only reports it.

The classification rests on three coupled tables, the accumulated firing rates, the per-class proportions, and the neuron-to-class assignments, each held as a single-row memory read one row at a time. The class of a sample is decided at the boundary from the spike counts. Each neuron is assigned to the class it has historically fired most for, the counts of all neurons sharing a class are summed and divided by the number of neurons assigned to that class so that classes with many neurons are not favoured, and the class with the largest normalized score wins. A second score weights each neuron's count by its class proportions before the same normalization, giving a proportion-weighted decision in parallel with the plain count-based one.

Rather than sweep all N neurons at the boundary, the readout accumulates these scores during the sample. On each output spike the proportion row of the winning neuron is added into a set of per-class accumulators, so that the argmax of the accumulators is the decision and their maximum divided by their sum is the confidence. All of these steps rely on division, again the costly operation. The boundary divisions, the two classification scores and, under learning, the table updates below, use the same sequential radix-2 divider as the normalization. The running confidence is instead evaluated on every timestep, where a full sequential divide would be too slow, so it uses a reduced-precision combinational divide that forms only the top few quotient bits, enough for a monitor value within a single cycle.

When learning is enabled, the tables are updated at the boundary and only for the neurons that spiked. Each spiking neuron's counts are accumulated into its rate row under the sample's label, its proportions are recomputed from the new rates, and its assignment is changed only if the leading proportion now points to

a different class. Restricting the update to the spiked rows keeps it cheap, and it leaves room for a further idea from Chapter 3. Because the spike counts already reflect how each neuron's threshold has moved, a threshold-driven leak of the rates would add almost no hardware, reusing the counts already collected here.

4.4.6 Control FSM.

A single 21-state control FSM sequences the datapath: a per-timestep schedule (leak, current-scan launch and wait, inhibition, threshold-and-fire, winner selection, and result registration) followed, when learning is enabled, by the trace and weight-update stages. At the sample boundary it starts prediction and, for learning, normalization and readout calibration in parallel before applying the combined threshold decay and signalling completion. Handshakes hold the FSM in the scan, STDP, prediction, normalization, and calibration wait states until the corresponding block finishes, while the AXI input-valid signal supplies back-pressure between timesteps. The schedule therefore grows with the number of active inputs and spikes rather than directly with M or N; adding neurons adds lanes rather than inference cycles, as used in Section 4.5. The learn-enable signal bypasses the trace, STDP, normalization, calibration, and threshold-decay states, so frozen inference and on-chip learning share the same datapath. Weights, readout tables, and the threshold master persist between samples, whereas working membrane and counter state is reset. The synthesized control hierarchy also includes the M-wide delayed-input steering, the N-wide winner mask, and the wide enables and multiplexers rather than only the state decoder; this is why it accounts for 36,421 LUTs, reported with top-level interconnect as approximately 37k LUTs.

4.4.7 Cycle Budget

Collecting the per-operation cycle counts in one place, Table 4.2 shows where a sample's time is spent and is the evidence behind the projection in Section 4.6. The two control phases omitted from the earlier budget, current-scan launch and result register/count/advance, add about two cycles per timestep. The profiled inference loop therefore costs about 21 cycles per timestep rather than 19. Cycle-accurate simulation measures averages of 5,409 cycles for inference and 16,268 for online learning; the individual rows below are rounded, activity-dependent averages. The wall-clock latencies reported below are larger because they also include DMA streaming, cache maintenance, polling, and readout overhead.

Table 4.2: Cycle budget per sample at $N = 400$, $T = 250$. Inference runs the per-timestep block; learning adds the trace decay each timestep and the STDP and normalization at the sample boundary.

Phase	Operation	Cycles
Per timestep (inference)	membrane leak	1
	current-scan launch	1
	synaptic currents	$\approx$ 15–16 (activity dependent)
	recurrent inhibition	1
	threshold and fire	1
	winner-take-all	1
	register, count, and advance	1
	profiled subtotal	≈ 21
Per timestep (learning)	pre-trace decay	2 ($\lceil M/N \rceil$)
Per sample (learning)	STDP weight update	$\approx$ 4,000
	weight normalization	$\approx$ 5,000
Per sample (both)	readout prediction	≈ 29
Measured total (inference)	complete sample	5,409 average
Measured total (learning)	complete sample	16,268 average

4.5 Results

4.5.1 Measurement Setup

The design was brought up on the ZCU102 board connected to a host PC, shown in Figure 4.5. The board attaches to the host over two USB links, a USB-JTAG link used to program and debug the device and a USB-UART link over which the software running on the processing system prints its results. The hardware was designed in Vivado, and the bitstream and the processing-system application were loaded onto the programmable logic and the processor with Vitis. During a run the on-chip software reports the inference and learning results over the UART with `xil_printf`, and the host records them.

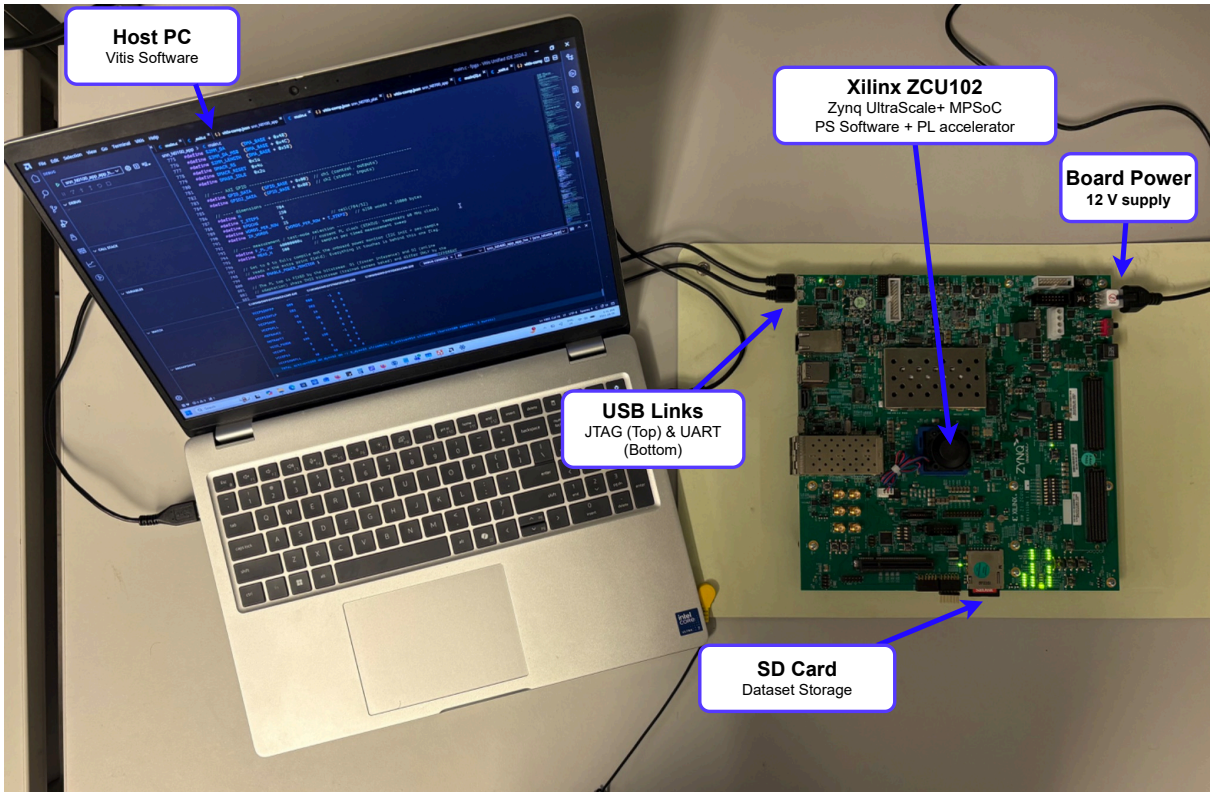


Figure 4.5: Measurement setup: the ZCU102 board connected to a host PC over USB-JTAG (programming and debug) and USB-UART (result readout).

Post-route timing at a 96.97 MHz target (a 10.31 ns period) gives a worst negative slack of -4.6 ns, so the confidence-monitor/readout-argmax critical path is approximately 14.9 ns. This corresponds to an estimated maximum frequency of approximately 67 MHz. The implemented design runs at 60 MHz, sourced from the PS clock PLLs, and closes timing with positive slack; all reported latencies and measurements use that operating frequency.

The reported latencies are accelerator time only, the streaming and on-chip computation of a sample. Poisson encoding runs on the PS and is pre-computed into DDR ahead of a run, so it is not included; performed on the fly on the ARM core it would be encode-bound at roughly 0.5 to 1 ms per sample, which is the reason it is pre-computed.

Power is measured on the board rather than estimated by a tool. The ZCU102 carries an INA226 current and voltage monitor on each supply rail, read over the PS I^2C bus. Because a single conversion is coarse and unsynchronized to a sample, each reported figure is the average power over a sustained back-to-back run of many samples, minus an idle baseline taken with the clock running but no samples issued, so that the

dynamic energy per sample is $(P_{\text{active}} - P_{\text{idle}}) \cdot t_{\text{sample}}$; this averaging pass runs separately from the timed loop so that the I^2C reads do not inflate the latency. Inference and learning are separated by the learn-enable flag on the same bitstream. The board draws about 2.6 W in total, but most of that is the ARM processing system and DDR, not the accelerator; the accelerator's own dynamic power sits almost entirely on the VCCINT (PL core) rail. Unless stated otherwise, all power and energy figures in this chapter are the VCCINT dynamic values.

4.5.2 Functional Validation

The design point used throughout the results is the $N = 400$ network trained on-chip, which reaches 86.2% on the MNIST test set, close to the 87% that Diehl and Cook report for a network of the same size [44]. Correctness is checked against a reference model that carries the same simplifications and quantization as the hardware, namely the lowest-index one-spike rule in place of the multinomial draw, the once-per-sample threshold leak, and the fixed-point widths of Section 4.3. Given identical input spikes, the FPGA and this reference model produce bit-identical results, which confirms that the hardware realizes the intended model exactly rather than approximating it.

4.5.3 Neuron-Count Sweep

The network was trained from scratch on-chip at $N \in \{100, 400, 700\}$ and evaluated in both inference and learning, both to place the design against the Diehl and Cook baseline and to expose the trade-offs of the fully parallel choice. Table 4.3 collects the results. Because the datapath is fully parallel, the inference decision time stays constant at 131 μs regardless of N, and the cost of that constant latency is resource use that grows with the network. A time-multiplexed design would sit at the opposite corner. If, for instance, one hundred processing units were shared across the neurons, running the $N = 400$ network would take about four passes and roughly four times the inference time, trading latency and throughput for a smaller datapath.

The same four-fold factor would apply to the switching energy, so a time-multiplexed $N = 400$ inference would spend on the order of 24 μJ, close to the 22 μJ of the parallel design; what a time-multiplexed design normally offers in return is lower static energy and lower utilization, at the price of throughput. On this platform the static-energy advantage does not appear, because the board power is dominated by the processing system and stays between 2.61 and 2.69 W across the whole sweep regardless of the accelerator. Learning behaves the same way between $N = 400$ and 700, where most of the added time comes from the

Table 4.3: On-chip neuron-count sweep, trained from scratch on-chip, against the Diehl and Cook baseline [44]. Latency, dynamic power, and energy are per sample; power and energy are the VCCINT dynamic figures averaged over a run, and the first samples of a run cost more time and energy as the network settles. The displayed accuracies are best observed values obtained with unequal training budgets, reported explicitly in the epochs column. The $N = 400$ LUT entry is from the sweep-specific placed build (147,222 LUTs, 53.7%); the final main implementation used 140,339 LUTs (51.2%), as reported separately in Section 4.5.4. The Diehl and Cook value at $N = 700$ is interpolated, since they report 100, 400, and 1,600 neurons.

	This work		D&C		Inference			Learning		
N	acc. (%)	epochs	acc. (%)	LUT (%)	lat. (μs)	pow. (mW)	E (μJ)	lat. (μs)	pow. (mW)	E (μJ)
100	79.2	2	82.9	23.9	131	46	6	350	66	23
400	86.2	3	87.0	53.7	131	168	22	292	226	66
700	84.8	5	88.2	80.6	131	298	39	287	404	116

STDP writes and the normalization of Section 4.4. A time-multiplexed learning datapath would not shrink the intermediate state, however. The traces, membranes, and other per-neuron variables still have to be held for every neuron, so the flip-flop and memory cost grows with N and can become the binding resource in its own right. The practical conclusion is that, on this design, adding neurons buys only the chance of higher accuracy, paid for in energy that scales with N and, under time-multiplexing, in throughput.

Several trends follow directly. Inference latency is flat at 131 μs because adding neurons adds parallel units rather than timesteps, so the per-sample cycle count, set by the T timesteps and the N-independent active-input scan, does not grow with the network. Learning latency is higher and falls slightly with N (350, 292, 287 μs); its elevated value at $N = 100$ comes mostly from the pre-trace decay, which folds the M inputs over the N multipliers in $\lceil M/N \rceil$ passes each timestep, eight passes at $N = 100$ against two at $N = 400$ and 700 (Section 4.4). Accelerator energy instead scales roughly linearly with N, from 6 to 22 to 39 μJ in inference, because each added unit is a multiplier and adder that switch every timestep. LUT utilization grows the same way, from 23.9 to 80.6% at about 260 LUTs per neuron, and the LUT is the limiting resource, with the device ceiling near $N = 700$–750, since $N = 800$ fails placement while BRAM and DSP stay comfortable.

Accuracy is not monotonic (79.2, 86.2, 84.8%), but the sweep used unequal budgets of two, three, and five epochs, respectively, so it is not a controlled accuracy-versus-size comparison. At $N = 100$ the first epoch gives 73.1% and the second raises it to 79.2%; the smaller network nevertheless remains below the $N = 400$ result. Conversely, the $N = 700$ network received the largest budget but did not surpass $N = 400$, so its lower accuracy cannot be explained simply as under-training.

The initially suspected cause was the deterministic lowest-index one-spike rule of Section 4.4, which could favour low-index neurons when several cross threshold together. That hypothesis was tested on-chip with

a learn-gated rotating WTA start index over five epochs. The modified design peaked at 84.27%, close to and slightly below the 84.8% baseline, and the following epoch reduced accuracy. Rotation therefore did not recover the $N = 700$ result, indicating that lowest-index tie-breaking is not the cause of the plateau. Its source remains unresolved; possible factors include the threshold-homeostasis parameters and how effectively the larger population is used under the present training budget.

The cost of a full on-chip training run follows from the same measurements. Training the $N = 400$ network from scratch used the whole 60,000-image set for three epochs; at the 292 μs learning latency those 180,000 samples take about 53 seconds of wall-clock time, and at 66 μJ per learning sample about 12 J of accelerator dynamic energy in total. For a device meant to adapt after deployment, this one-off training cost is arguably more telling than the cost of a single sample, and it is small enough to be paid on the device itself.

4.5.4 Resource Utilization

The final main implementation at $N = 400$ uses 140,339 LUTs (51.2% of the ZU9EG), approximately 74k flip-flops (14%), 803 DSP slices (32%), and 209 BRAMs (23%); the LUT is the tightest resource and sets the device ceiling seen in the sweep. For a chapter about whether learning can be afforded on chip, the more telling number is how that area divides between the inference spine, which is present in both modes, and the parts that exist only for plasticity. Table 4.4 breaks the utilization down by module. The learning engine accounts for about 43k LUTs, roughly 30% of the logic, which is the direct area price of on-chip plasticity; the inference spine (dominated by the N neuron lanes and their DSPs) and the control sequencer take most of the rest, while the readout head and its memories are almost free.

Table 4.4: Per-module resource utilization at $N = 400$, from the synthesized hierarchical report. The learning engine is the cost of on-chip plasticity; the inference spine runs in both modes. LUT figures include logic shared between modules and so sum slightly above the synthesized 146,023-LUT total. The separately reported final main implementation uses 140,339 LUTs, approximately 74k FFs, 803 DSPs, and 209 BRAM tiles.

Module	LUT	FF	DSP	BRAM
Inference spine (neuron lanes, weight memory, θ, currents, WTA)	60.2k	42.9k	800	201
Learning engine (traces, STDP weight engine, normalizer, calibration)	43.2k	17.0k	3	1
Readout head (counters, rate/prop/assign tables, predict, confidence)	8.8k	7.9k	0	7
Control FSM and interconnect	37.0k	1.8k	0	0
Total (synthesis)	146k	69.6k	803	209

4.5.5 Power and Energy

Figure 4.6 shows the per-rail power breakdown at $N = 400$, confirming that the accelerator's dynamic power sits on the VCCINT rail while the board total is carried by the processing-system rails (Section 4.5.1).

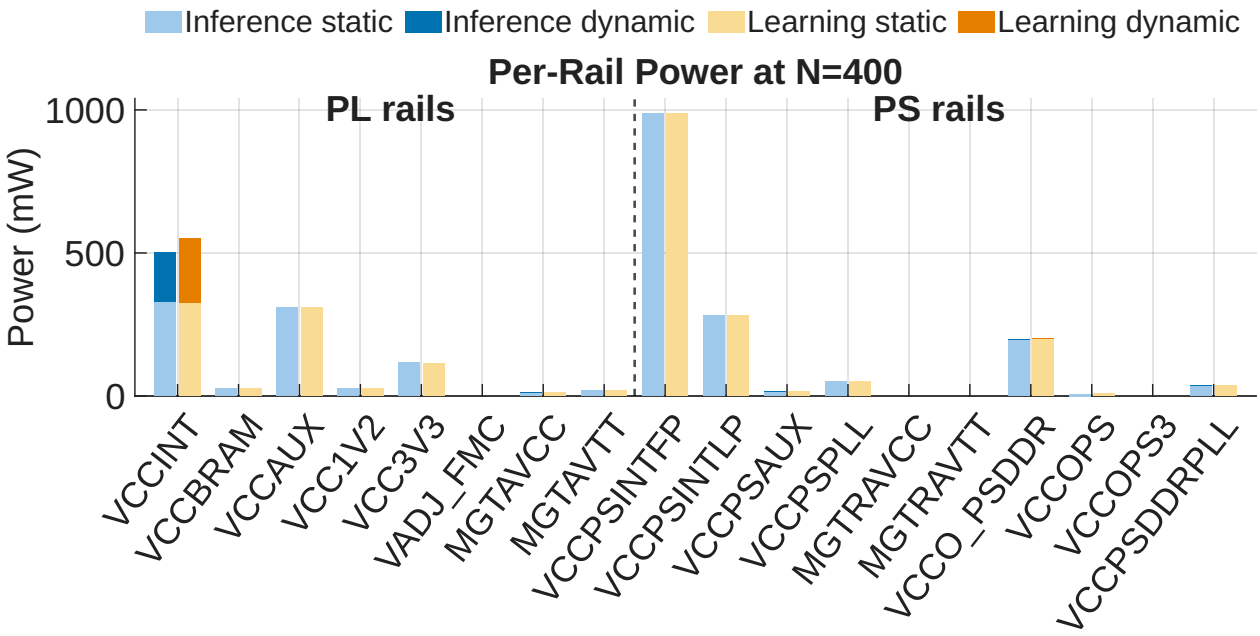


Figure 4.6: Per-rail power at $N = 400$, inference versus learning, with the static (idle) base and the dynamic component stacked. The accelerator's dynamic power sits on VCCINT; the board total is dominated by the processing-system static rails.

At $N = 400$ the VCCINT dynamic power is 168 mW in inference and 226 mW in learning, learning drawing $+35\%$ for the additional trace updates, STDP, normalization, and calibration switching, which gives 22 and 66 μJ per sample. The $3\times$ learning-to-inference energy ratio decomposes as $2.22\times$ in time and $1.35\times$ in power.

4.5.6 Comparison with Prior Work

Tables 4.5 and 4.6 separate the architectural causes of resource use from the resulting accuracy, latency, and energy. This is necessary because a small LUT or BRAM total may describe a smaller network, a time-multiplexed set of processing elements, narrower fixed weights, inference-only hardware, or only part of the complete system. Absolute resource counts without those qualifications are therefore not a like-for-like area comparison.

The 209 BRAMs in this work are primarily a bandwidth cost rather than raw storage capacity. The 784×400

Table 4.5: Architecture and resource comparison for the implemented MNIST configurations. The network column gives input–hidden–output dimensions where applicable; E, I, and T denote excitatory, inhibitory, and teacher neurons. "n.r." means not reported.

Reference	Platform	Input	Network	Parallel hardware	Weight format	LUT	BRAM
[90]	Virtex-6 (40 nm)	28×28	784+6I–800+1I	32 LAUs	4-bit	97,287	$\sim 340^a$
[85]	Zynq-7000 (28 nm)	28×28	784–100E	14×10 array	10-bit	78,053	n.r.[b]
[64]	Virtex-7 (28 nm)	28×28	784–100E–100I	12 PEs	FP16	n.r.	n.r.
[87]	Artix-7 (28 nm)	28×28	784–400E	400 neuron lanes	5-bit	29,145	45
[86]	Virtex-7 (28 nm)	28×28	784–400E–400I	32 cores, 4 reads/core	25-bit fixed[c]	54,313	101
[88]	Zynq-7045 (28 nm)	28×28	784–2048–100	32 hidden + 4 output CUs	2-bit hidden[d]	16,813	129
[66]	Zynq-7045 (28 nm)	28×28	784–256	4 NPTs	16-bit	10,052	131
[89]	ZC706 (28 nm)	12×12	144+10T–100E–100I–10	16×10 PEs	16-bit integer	n.r.	~ 330
This work	ZCU102 (16 nm)	28×28	784–400E	400 neuron lanes	16-bit	**140,339**	**209**

[a]The 34 entries called BRAMs in [90] are logical memory banks rather than physical FPGA primitives; the physical-resource comparison in [86] reports approximately 340 36-Kb BRAMs for that implementation. [b]The zero-BRAM entry in the compute-array comparison of [85] excludes the BRAM-I, BRAM-S, and BRAM-E memories shown in its complete architecture, whose total is not reported. [c]Reference [86] reports a 25-bit datapath format but does not isolate the stored weight width. [d]Reference [88] regenerates the fixed hidden weights at 2-bit precision; the precision of the plastic output weights is not stated. Its LUT total includes 15,621 logic LUTs and 1,192 LUTRAMs.

matrix of 16-bit weights contains 5,017,600 bits, which would need only 137 36-Kb blocks if it could be packed solely for capacity. Instead, the implementation uses one independently addressable bank per output neuron: 400 RAMB18 primitives, reported as 200 BRAM36-equivalent blocks. Each bank stores $784 \times 16 = 12{,}544$ bits in an 18,432-bit primitive, or about 68% of its capacity. This organization provides all 400 neuron lanes with a concurrent weight read and supports a selected bank's read–modify–write path during STDP. The remaining nine BRAM36 equivalents hold other state and readout memories. Thus, the difference between the capacity-only minimum and the implemented total is the cost of banking for parallel access, not an unexplained increase in model storage.

The LUT comparison leads to a different conclusion. At the same 400-neuron physical parallelism, the neuron array in [87] uses 23,885 LUTs, whereas the neuron array in this work uses 33,636 LUTs, a factor of 1.41. The complete accelerators use 29,145 and 140,339 LUTs, respectively, so most of the absolute difference lies outside the basic neuron array. The inference-only implementation in [87] uses frozen 5-bit weights, shift-based leakage, a shared random-number generator, and no on-chip plasticity. This work additionally contains writable weight-bank selection, pre- and post-synaptic traces, STDP and normalization, adaptive threshold state, calibration, confidence/readout logic, and the control and interconnect needed to sequence them, as quantified in Table 4.4. Reference [87] includes its input spike generator, while this work receives pre-encoded spike rasters, so even these totals do not describe identical system boundaries.

Device-normalized utilization further changes the interpretation of the absolute numbers. This work uses

Table 4.6: Performance comparison with representative FPGA STDP accelerators. Accuracies are for MNIST at each reference's stated configuration; latency and energy are per image. This work's energy is measured on the board, whereas prior values are estimates or do not state a measurement method. "n.r." means not reported and "—" means not applicable. This work excludes Poisson-encoding time.

			Inference		Learning	
Reference	On-chip learning	Accuracy (%)	latency	energy	latency	energy
[44]	software baseline	82.9–95	—	—	—	—
[90]	unsupervised	89.1	8.4 s	1.13 J	16.8 s	1.34 J
[85]	unsupervised	83.4	n.r.	n.r.	6.1 ms	6.65 mJ
[64]	unsupervised	85.3	3.15 ms	5.04 mJ	16.3 ms	26.3 mJ
[87]	none (offline)	74.0	215 μs	13 mJ	—	—
[86]	unsupervised	88.6	18 ms	1.39 mJ	19 ms	1.74 mJ
[88]	supervised	93.0	33.3 ms	n.r.	44.4 ms	n.r.
[66]	hybrid	95.1	0.37 ms	0.35 mJ	0.74 ms	0.70 mJ
[89]	supervised	82.2	500 ms	n.r.	500 ms	n.r.
This work	unsupervised	86.2	**131 μs**	**22 μJ**	**292 μs**	**66 μJ**

$209/912 = 22.9\%$ of its BRAM capacity, compared with 23.7% in [88], 24.0% in [66], 32% in [87], and 60.55% in [89]. Similarly, the 140,339 LUTs in this work occupy 51.2% of the ZU9EG, while the much smaller absolute count of 29,145 LUTs in [87] is reported as 55% of its device. The present design is therefore large in absolute LUT count, but not anomalously large as a fraction of the selected FPGA; its area is the cost of combining 400 physical neuron lanes with a complete on-chip learning and readout path.

The performance consequence of that choice is summarized in Table 4.6. For high-throughput image processing, time-multiplexed designs with supervised or hybrid readout can reach higher accuracy, but for low-latency on-device learning the relevant quantities are per-sample latency, per-sample energy, and whether learning runs on chip without labels. The fully parallel datapath gives this work the lowest reported per-image latency in the table (131 μs, or 7,616 samples/s), while retaining unsupervised on-chip STDP. Its 7,616-sample/s figure applies to pre-encoded spike rasters streamed from DDR and excludes Poisson encoding; an on-the-fly ARM encoder would add approximately 0.5–1 ms per sample, while a deployed streaming system could place the random-number-and-compare encoder in programmable logic [91]. The 86.2% accuracy is the measured operating point for the three-epoch $N = 400$ run. The board uses a 16 nm device while most prior designs use 28 nm or older devices, and the energy boundaries and estimation methods differ; the energy entries should therefore be read as order-of-magnitude rather than like-for-like comparisons.

4.6 Discussion

The measured learning cost lets the algorithmic savings of the preceding chapters be put in concrete hardware terms, even though those mechanisms are not yet in the RTL. At $N = 400$ a learning sample takes 292 μs against 131 μs for inference, so the learning-specific work is 161 μs. Almost all of it is weight traffic. The STDP update (about 4,000 cycles) and the boundary normalization (about 5,000 cycles) of Section 4.4 together come to roughly 9,000 cycles, close to 150 μs, about 93% of the learning-specific time.

This is exactly the traffic the selective-adaptation rule of Chapter 2 removes. Its confidence-gated dual-path rule commits a weight update only on confident samples, cutting the weight reads and writes by up to 80%, and adding the selectivity gate, which stops updating neurons that have already specialized, raises the cut to 92%. Applied to the 150 μs of weight traffic, the learning latency would fall from 292 to about 172 μs with the dual-path rule and 154 μs with the selectivity gate. At the same power this lowers the learning energy from 66 to about 39 and 35 μJ per sample, reductions of 41% and 47%, while the same rules improve accuracy on most of the adaptation tasks of Chapter 2. Because these mechanisms are characterized algorithmically and not yet built, the figures are projections from the measured cost rather than measured results.

The imbalance chapter offers a hardware answer to the trade-off the sweep exposed, where a larger network needs either more parallel units or a loss of throughput to time-multiplexing. Its topographic window organizes the neurons on a two-dimensional map so that each input excites only a local neighbourhood, about 9×9 neurons, close to the $N = 100$ point. A network with the capacity of $N = 400$ could then be evaluated with only about one hundred parallel units, since only the active window is computed, keeping the utilization, energy, and latency of the $N = 100$ design at a small cost in accuracy. The window settles quickly, within under a tenth of the timesteps, so the inference latency stays close to that of $N = 100$.

More broadly, the asymmetry among the three plastic states, the weight matrix being large and slow while the thresholds and rate statistics are small and fast, points toward a mixed-memory implementation for an ASIC, a dense read-optimized store for the weights and a small frequently-written memory for the thresholds and rates. Realizing that split is left to future work.

4.6.1 Limitations

Three limitations bound the present results. First, the device ceiling is set by logic placement: the $N =$ 700 design already uses 80.6% of the LUTs, the practical limit is approximately $N = 700$–750, and $N =$

800 fails placement even though its BRAM and DSP demands remain within the device. Larger networks therefore require a larger FPGA or a partially time-multiplexed datapath. Second, the full-board power of approximately 2.6 W is dominated by the ARM processing system and DDR rather than by the SNN logic. Reporting the VCCINT dynamic component isolates the accelerator, but a leaner host or an ASIC would be needed to demonstrate the same efficiency at the complete-system level. Finally, the neuron-count sweep used unequal budgets of two, three, and five epochs, so it shows implemented operating points rather than a controlled accuracy-versus-size comparison. The negative rotating-WTA result also leaves the $N = 700$ plateau unexplained. More broadly, the accelerator is an optimized fixed-point realization validated at the prediction level, not a bit-exact reproduction of the original floating-point model.

4.7 Chapter Summary

This chapter translated the unsupervised STDP network into a complete FPGA accelerator that supports both frozen inference and on-chip learning. Its architecture combines N parallel neuron lanes with neuron-banked BRAM, allowing each active input to broadcast one weight row to all neurons in parallel, while sparse current accumulation, trace scanning, STDP updates, normalization, and readout calibration avoid work on inactive state. A shared control FSM sequences the inference and learning phases, and the $Q0.16$ weight format retains sufficient headroom for the graded depression update even though inference accuracy itself saturates at substantially lower precision.

For the principal $N = 400$ implementation, the network reaches 86.2% MNIST accuracy after three epochs of on-chip training. With pre-encoded spike rasters streamed from DDR, inference takes 131 μs and 22 μJ of VCCINT dynamic energy per sample, while learning takes 292 μs and 66 μJ. Post-route analysis gives an estimated maximum frequency of approximately 67 MHz, and the measured system operates at 60 MHz with positive timing slack. The final implementation uses 140,339 LUTs, 803 DSP slices, and 209 BRAM tiles. The neuron-count sweep confirms the central parallelism trade-off: inference latency remains constant as N grows, whereas logic use and dynamic energy increase approximately linearly, producing a practical device ceiling near $N = 700$–750. Together, these results demonstrate that unsupervised adaptation can be executed on-chip at sub-millisecond latency and tens of microjoules per sample, while also identifying weight traffic, logic scaling, host-side encoding, and unequal training budgets as the main opportunities and limits for the selective-adaptation and class-imbalance mechanisms developed in the preceding chapters.

5 Conclusion and Future Work

5.1 Conclusion

This thesis investigated how an unsupervised STDP-SNN can remain adaptive without treating all plastic state as continuously writable. The motivating application is an implantable biosignal processor that must operate under strict energy and memory constraints, learn from a changing input stream, and preserve representations of rare events. The resulting design principle is that plasticity should be allocated according to the reliability of the available learning signal and the state of the neuron population. The thesis evaluated this principle through selective online adaptation, imbalance-robust training and inference, and a measured FPGA realization of the baseline learning network.

Chapter 2 separated the persistent state of the baseline network into synaptic weights, adaptive thresholds, and firing-rate statistics. Freeze ablations showed that these states converge on different timescales and should not share one update schedule. A confidence-conditioned dual-path rule therefore commits weights on confident samples, updates the smaller homeostatic and assignment states on non-confident samples, and abstains in an intermediate band. A sample-relative selectivity gate further prevents already specialized neurons from receiving unnecessary weight commits while allowing them to become plastic again when their selectivity declines. Across the controlled covariate shifts, the combined policy improved mean minority sensitivity relative to both frozen and all-state adaptation and reduced weight-commit traffic by approximately 80% during active adaptation and 92% at steady state.

Chapter 3 addressed the distinct problem of training under severe class imbalance. Per-neuron analysis showed that majority-dominated WTA competition reassigns minority specialists and creates redundant majority representations. Four local, ratio-agnostic mechanisms were introduced to protect competition, inhibitory authority, assignment memory, and weight consolidation without resampling or an estimate of the imbalance ratio. At an imbalance ratio of 1:1000, these mechanisms increased minority sensitivity from 41% to 72% while maintaining 85% specificity. A topographic organization then enabled first-spike-anchored, window-restricted inference. The selected 9×9 window retained 68% sensitivity and reduced modeled inference energy from 6.7 to $1.8\,\mu$J.

Chapter 4 established the implementation feasibility and measured cost of the common V2 learning baseline. The fixed-point FPGA accelerator performs adaptive LIF dynamics, WTA competition, trace-based STDP, normalization, and readout calibration in programmable logic. At the principal 400-neuron operating point,

it achieved 86.2% MNIST accuracy after three epochs of on-chip training. Inference took 131 μs and 22 μJ of programmable-logic dynamic energy per sample, while learning took 292 μs and 66 μJ. These measurements identify synaptic update and normalization traffic as central costs of online learning and provide the hardware context for the update reductions developed in the preceding chapters.

Taken together, the results show that online plasticity need not be monolithic. A network can preserve slow synaptic representations, direct fast correction to smaller neuron-level state, protect rare-class specialists, and restrict computation to the neurons relevant to the current input. The algorithmic mechanisms and FPGA baseline were deliberately evaluated as separate studies: the selective-adaptation and imbalance controls have not yet been integrated into the measured accelerator, the energy reductions in Chapter 3 are modeled rather than measured, and MNIST is a controlled benchmark rather than a clinical biosignal. Within these boundaries, the thesis demonstrates a consistent path toward adaptive SNN processors whose memory activity and computation follow the needs of the current sample.

5.2 Statement on Contributions

Two papers arising from the work in this thesis were accepted to the 2026 IEEE Biomedical Circuits and Systems Conference (BioCAS 2026).

[1] D. Valipour and H. Kassiri, "Memory-Efficient Unsupervised Online Adaptation in SNNs via Confidence- and Selectivity-Gated STDP," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, 2026, accepted.

[2] D. Valipour and H. Kassiri, "Imbalance-Robust, Ratio-Agnostic, Energy-Efficient STDP in Topographic SNNs for Rare Event Detection," in *Proc. IEEE Biomed. Circuits Syst. Conf. (BioCAS)*, 2026, accepted.

The first paper presents the state-timescale analysis, confidence-conditioned dual-path adaptation, and neuron-selectivity gate developed in Chapter 2. The second presents the diagnosis of population-allocation failure, four local imbalance-robust plasticity mechanisms, and topographic window-restricted inference developed in Chapter 3. The thesis expands these studies through a common system motivation and network background, explicit discussion of their validation boundaries and possible interaction, and the measured FPGA baseline in Chapter 4.

5.3 Future Work

The first direction is to integrate the controls of Chapters 2 and 3 in one long-duration experiment. The present adaptation study starts from a pretrained model and introduces covariate shift, whereas the imbalance study trains from scratch under a controlled class ratio. A combined evaluation should vary both the feature distribution and event prevalence, quantify the effect of imperfect pseudo-labels, and determine whether confidence routing, specialist protection, and weight consolidation remain compatible over prolonged adaptation.

The second direction is validation on clinical biosignals. Long-duration EEG, ECG, or other rare-event recordings should replace the controlled MNIST benchmark while preserving patient-independent training and patient-specific evaluation. Such a study should report event sensitivity and specificity together with false alarms per unit time, detection latency, calibration drift, and performance across patients. It should also determine whether the confidence and selectivity statistics used here remain reliable when recordings contain artifacts, nonstationarity, and uncertain labels.

The third direction is a hardware implementation of the selective-adaptation and imbalance mechanisms. Adding the confidence controller, per-neuron selectivity gate, state-conditioned inhibition, assignment retention, consensus scaling, and topographic window to the FPGA design would allow their area, latency, memory traffic, and energy effects to be measured rather than projected. The same platform could evaluate state-aware temporal termination, in which a mature and selective first winner ends an input presentation only after a minimum observation interval and a sustained confidence or winner-margin condition.

Finally, an implantable ASIC should separate state according to its write behavior. The large, slowly updated weight matrix is a candidate for dense non-volatile or read-optimized memory, whereas thresholds, traces, and firing-rate statistics require a smaller, high-endurance store. Integrating the signal front end, spike encoder, learning processor, and readout would also remove the processing-system and DDR overhead of the FPGA prototype. Silicon measurements on such a system are needed to establish endurance, area, thermal behavior, and total energy per decision under a realistic continuous biosignal workload.

Bibliography

[1] M. Taghadosi and H. Kassiri, "A calibration-free energy-efficient IC for link-adaptive real-time energy storage optimization of CM inductive power receivers," *IEEE Journal of Solid-State Circuits*, vol. 57, no. 3, pp. 793–802, 2022.

[2] A. Soliman, Y. Khazaei, A. Shaghayegh, and H. Kassiri, "A non-uniform-sparse ultrasonic phased array for channel-efficient powering of neural implants," in *2026 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2026. Accepted for publication.

[3] M. Sayedi, M. Korosteliov, S. Khakneshin, and H. Kassiri, "Electrode-to-antenna activity-adaptive cortical monitoring via stat-informed DQ-LC-ADC and an event-driven compressive UWB transmitter," in *2026 IEEE European Solid-State Electronics Research Conference (ESSERC)*, 2026.

[4] H. Kassiri, M. T. Salam, M. R. Pazhouhandeh, N. Soltani, J. L. Perez Velazquez, P. L. Carlen, and R. Genov, "Rail-to-rail-input dual-radio 64-channel closed-loop neurostimulator," *IEEE Journal of Solid-State Circuits*, vol. 52, no. 11, pp. 2793–2810, 2017.

[5] R. Salahi and H. Kassiri, "NMM-based patient-specific temporally-adaptive stimulation optimization for seizure control," in *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2024.

[6] S. Nabavi and H. Kassiri, "Toward prediction-driven time-adaptive seizure management using a WC-NMM network," in *2025 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2025.

[7] D. Koudoumnakis and H. Kassiri, "An electrode-coupled patient-adaptive cortical twin with self-driven seizure dynamics for predictive neurostimulation," in *2026 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2026. Accepted for publication.

[8] T. Yousefi, G. Zoidl, and H. Kassiri, "A highly-scalable Poisson-coded retinal optogenetic stimulator with fully-analog ED-based adaptive spike detection and closed-loop calibration," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 18, no. 6, pp. 1253–1267, 2024.

[9] H. Kassiri, S. Tonekaboni, M. T. Salam, N. Soltani, K. Abdelhalim, J. L. Perez Velazquez, and R. Genov, "Closed-loop neurostimulators: A survey and a seizure-predicting design example for intractable epilepsy treatment," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 11, no. 5, pp. 1026–1040, 2017.

[10] H. Kassiri, A. Muneeb, R. Salahi, and A. Dabbaghian, "Closed-loop implantable neurostimulators for individualized treatment of intractable epilepsy: A review of recent developments, ongoing challenges, and future opportunities," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 18, no. 6, pp. 1268–1295, 2024.

[11] A. M. Sodagar, Y. Khazaei, M. Nekoui, and M. Shaeri, "Real-time, neural signal processing for high-density brain-implantable devices," *Bioelectronic Medicine*, vol. 11, p. 17, 2025.

[12] M. Sayedi and H. Kassiri, "A neural-ADC-compatible fully-dynamic lossless adaptive resolution compression technique for energy-constrained bio-signal recording," in *2023 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2023.

[13] M. Sayedi and H. Kassiri, "Direct neural ADC via level-crossing quantization: Feasibility, trade-offs, and design guidelines," in *2026 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 4458–4462, 2026.

[14] M. Namavar and A. M. Sodagar, "A two-phase, single-comparator, comparison-reference-free level crossing ADC for time-sparse applications," in *2025 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2025.

[15] M. T. Salam, H. Kassiri, N. Soltani, H. He, J. L. Perez Velazquez, and R. Genov, "Tradeoffs between wireless communication and computation in closed-loop implantable devices," in *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1838–1841, 2016.

[16] M. Sayedi and H. Kassiri, "Activity-adaptive architectures for energy-efficient scalable neural recording microsystems: A review of current and future directions," in *2022 20th IEEE Interregional NEWCAS Conference (NEWCAS)*, 2022.

[17] A. Dabbaghian and H. Kassiri, "Energy-efficient automated seizure detection in wearable/implantable BCIs: Motivations, methods, and example implementation," in *2025 IEEE International Symposium on Multiple-Valued Logic (ISMVL)*, pp. 57–62, 2025.

[18] T. Zhan, S. Fatmi, S. Guraya, and H. Kassiri, "A resource-optimized VLSI implementation of a patient-specific seizure detection algorithm on a custom-made 2.2-cm^2 wireless device for ambulatory epilepsy

diagnostics," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 6, pp. 1175–1185, 2019.

[19] A. Dabbaghian and H. Kassiri, "Modular DR- and CMR-boosted artifact-resilient EEG headset with distributed pulse-based feature extraction and neuro-inspired boosted-SVM classifier," *IEEE Journal of Solid-State Circuits*, vol. 60, no. 3, pp. 921–933, 2025.

[20] S.-A. Huang, K.-C. Chang, H.-H. Liou, and C.-H. Yang, "A 1.9-mw SVM processor with on-chip active learning for epileptic seizure control," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 2, pp. 452–464, 2020.

[21] Y.-Y. Hsieh, Y.-C. Lin, and C.-H. Yang, "A 96.2-nj/class neural signal processor with adaptable intelligence for seizure prediction," *IEEE Journal of Solid-State Circuits*, vol. 58, no. 1, pp. 167–176, 2023.

[22] C.-W. Tsai, R. Jiang, L. Zhang, M. Zhang, and J. Yoo, "Seizure-cluster-inception CNN (SciCNN): A patient-independent epilepsy tracking SoC with 0-shot-retraining," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 17, no. 6, pp. 1202–1213, 2023.

[23] J. Liu, X. Liu, X. Wang, Z. Xie, C. Guo, Z. Zhong, J. Fan, H. Qiu, Y. Xu, H. Qin, Y. Long, Y. Zhou, Z. Shen, L. Zhou, L. Chang, S. Liu, S. Lin, C. Wang, and J. Zhou, "A high accuracy and ultra-energy-efficient zero-shot-retraining seizure detection processor," *IEEE Journal of Solid-State Circuits*, vol. 59, no. 11, pp. 3549–3565, 2024.

[24] A. Muneeb, S. Mehrotra, and H. Kassiri, "A 9.5-ms-latency 6.2-μJ/inference spiking CNN for patient-specific seizure detection," in *2023 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 1–5, 2023.

[25] A. Muneeb and H. Kassiri, "Customized development and hardware optimization of a fully-spiking SNN for EEG-based seizure detection," in *2024 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pp. 1–5, 2024.

[26] C. Fang, C. Wang, S. Zhao, F. Tian, J. Yang, and M. Sawan, "A 510-μW 0.738-mm^2 6.2-pj/SOP online learning multi-topology SNN processor with unified computation engine in 40-nm CMOS," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 17, no. 3, pp. 507–520, 2023.

[27] C. Frenkel and G. Indiveri, "ReckOn: A 28nm sub-mm^2 task-agnostic spiking recurrent neural network processor enabling on-chip learning over second-long timescales," in *2022 IEEE International Solid-State Circuits Conference (ISSCC)*, 2022.

[28] A. Muneeb and H. Kassiri, "Energy-efficient spiking-CNN-based cross-patient seizure detection," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2023.

[29] A. Dabbaghian and H. Kassiri, "An 8-channel ambulatory EEG recording IC with in-channel fully-analog real-time motion artifact extraction and removal," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 17, no. 5, pp. 999–1009, 2023.

[30] S. Nabavi and H. Kassiri, "Rethinking ETI sensing: Analysis, experimental proof, and circuit insights for in-band, magnitude-only motion tracking in EEG," in *2026 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2026.

[31] M. R. Karimi and H. Kassiri, "A multi-feature nonlinear-SVM seizure detection algorithm with patient-specific channel selection and feature customization," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2020.

[32] M. F. Minhas, R. V. W. Putra, F. Awwad, O. Hasan, and M. Shafique, "Continual learning with neuromorphic computing: Foundations, methods, and emerging applications," *IEEE Access*, vol. 13, pp. 124824–124873, 2025.

[33] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[34] 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.

[35] P. Panda, J. M. Allred, S. Ramanathan, and K. Roy, "ASP: Learning to forget with adaptive synaptic plasticity in spiking neural networks," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 8, no. 1, pp. 51–64, 2018.

[36] J. M. Allred and K. Roy, "Controlled forgetting: Targeted stimulation and dopaminergic plasticity modulation for unsupervised lifelong learning in spiking neural networks," *Frontiers in Neuroscience*, vol. 14, p. 7, 2020.

[37] D. I. Antonov, K. V. Sviatov, and S. Sukhov, "Continuous learning of spiking networks trained with local rules," *Neural Networks*, vol. 155, pp. 512–522, 2022.

[38] Z. Mei, M. Barnell, and Q. Qiu, "Unsupervised adaptation of spiking networks in a gradual changing environment," in *2022 IEEE High Performance Extreme Computing Conference*, pp. 1–7, 2022.

[39] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, p. 27, 2019.

[40] G. Aguiar, B. Krawczyk, and A. Cano, "A survey on learning from imbalanced data streams: Taxonomy, challenges, empirical study, and reproducible experimental framework," *Machine Learning*, vol. 113, pp. 4165–4243, 2024.

[41] M. Horowitz, "1.1 computing's energy problem (and what we can do about it)," in *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers*, pp. 10–14, 2014.

[42] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[43] 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.

[44] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 9, p. 99, 2015.

[45] H. Hazan, D. J. Saunders, H. Khan, D. Patel, D. T. Sanghavi, H. T. Siegelmann, and R. Kozma, "BindsNET: A machine learning-oriented spiking neural networks library in python," *Frontiers in Neuroinformatics*, vol. 12, p. 89, 2018.

[46] 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.

[47] M. Sayedi and H. Kassiri, "Statistically-informed LC-ADC for energy-efficient input-adaptive wireless neural interfaces," in *2026 IEEE Interregional NEWCAS Conference (NEWCAS)*, 2026.

[48] F. Ottati, C. Gao, Q. Chen, G. Brignone, M. R. Casu, J. K. Eshraghian, and L. Lavagno, "To spike or not to spike: A digital hardware perspective on deep learning acceleration," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 13, no. 4, pp. 1015–1025, 2023.

[49] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological Cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[50] R. Naderi, A. Rezaei, M. Amiri, and H. Peremans, "Unsupervised post-training learning in spiking neural networks," *Scientific Reports*, vol. 15, p. 17647, 2025.

[51] G. J. Goodhill and H. G. Barrow, "The role of weight normalization in competitive learning," *Neural Computation*, vol. 6, no. 2, pp. 255–269, 1994.

[52] L. Prechelt, "Early stopping—but when?," in *Neural Networks: Tricks of the Trade* (G. B. Orr and K.-R. Müller, eds.), vol. 1524 of *Lecture Notes in Computer Science*, pp. 55–69, Berlin, Heidelberg: Springer, 1998.

[53] D. Valipour and H. Kassiri, "Memory-efficient unsupervised online adaptation in SNNs via confidence- and selectivity-gated STDP," in *2026 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2026. Accepted for publication.

[54] S. Fusi, P. J. Drew, and L. F. Abbott, "Cascade models of synaptically stored memories," *Neuron*, vol. 45, no. 4, pp. 599–611, 2005.

[55] M. K. Benna and S. Fusi, "Computational principles of synaptic memory consolidation," *Nature Neuroscience*, vol. 19, pp. 1697–1706, 2016.

[56] F. Zenke, B. Poole, and S. Ganguli, "Continual learning through synaptic intelligence," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995, 2017.

[57] J. M. Brader, W. Senn, and S. Fusi, "Learning real-world stimuli in a neural network with spike-driven synaptic dynamics," *Neural Computation*, vol. 19, no. 11, pp. 2881–2912, 2007.

[58] M. Noack, J. Partzsch, C. G. Mayr, S. Hänzsche, S. Scholze, S. Höppner, G. Ellguth, and R. Schüffny, "Switched-capacitor realization of presynaptic short-term-plasticity and stop-learning synapses in 28 nm CMOS," *Frontiers in Neuroscience*, vol. 9, p. 10, 2015.

[59] N. Frémaux and W. Gerstner, "Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules," *Frontiers in Neural Circuits*, vol. 9, p. 85, 2016.

[60] E. M. Izhikevich, "Solving the distal reward problem through linkage of STDP and dopamine signaling," *Cerebral Cortex*, vol. 17, no. 10, pp. 2443–2452, 2007.

[61] M. L. L. R. Barry and W. Gerstner, "Fast adaptation to rule switching using neuronal surprise," *PLoS Computational Biology*, vol. 20, no. 2, p. e1011839, 2024.

[62] F. M. Quintana, F. Perez-Peña, P. L. Galindo, E. O. Neftci, E. Chicca, and L. Khacef, "ETLP: Event-based three-factor local plasticity for online learning with neuromorphic hardware," *Neuromorphic Computing and Engineering*, vol. 4, no. 3, p. 034006, 2024.

[63] Z. Dong and W. He, "Astrocyte-gated multi-timescale plasticity for online continual learning in deep spiking neural networks," *Frontiers in Neuroscience*, vol. 19, p. 1768235, 2026.

[64] S. Li, Z. Zhang, R. Mao, J. Xiao, L. Chang, and J. Zhou, "A fast and energy-efficient SNN processor with adaptive clock/event-driven computation scheme and online learning," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 4, pp. 1543–1552, 2021.

[65] C. Sun, H. Sun, J. Xu, J. Han, X. Wang, X. Wang, Q. Chen, Y. Fu, and L. Li, "An energy efficient STDP-based SNN architecture with on-chip learning," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 12, pp. 5147–5158, 2022.

[66] H. Wang, Z. He, T. Wang, J. He, X. Zhou, Y. Wang, L. Liu, N. Wu, M. Tian, and C. Shi, "TripleBrain: A compact neuromorphic hardware core with fast on-chip self-organizing and reinforcement spike-timing-dependent plasticity," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 16, no. 4, pp. 636–650, 2022.

[67] F. T. Zohora, V. Karia, N. Soures, and D. Kudithipudi, "Probabilistic metaplasticity for continual learning with memristors in spiking networks," *Scientific Reports*, vol. 14, p. 29496, 2024.

[68] A. Amirshahi and M. Hashemi, "ECG classification algorithm based on STDP and R-STDP neural networks for real-time monitoring on ultra low-power personal wearable devices," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 6, pp. 1483–1493, 2019.

[69] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, "Memory devices and applications for in-memory computing," *Nature Nanotechnology*, vol. 15, pp. 529–544, 2020.

[70] A. Shahbazinia, J. Dan, J. A. Miranda, G. Ansaloni, and D. Atienza, "Personalization on a budget: Minimally-labeled continual learning for resource-efficient seizure detection," 2025. arXiv:2509.13974.

[71] J. Liu, J. Fan, Z. Zhong, H. Qiu, J. Xiao, Y. Zhou, Z. Zhu, G. Dai, N. Wang, Q. Liu, Y. Xie, H. Liu, L. Chang, and J. Zhou, "An ultra-low power reconfigurable biomedical AI processor with adaptive learning for versatile wearable intelligent health monitoring," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 17, no. 5, pp. 952–967, 2023.

[72] D. Valipour and H. Kassiri, "Imbalance-robust, ratio-agnostic, energy-efficient STDP in topographic SNNs for rare event detection," in *2026 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2026. Accepted for publication.

[73] A. Cano and B. Krawczyk, "ROSE: Robust online self-adjusting ensemble for continual learning on imbalanced drifting data streams," *Machine Learning*, vol. 111, pp. 2561–2599, 2022.

[74] Q. Zhou, C. Ren, and S. Qi, "An imbalanced R-STDP learning rule in spiking neural networks for medical image classification," *IEEE Access*, vol. 8, pp. 224162–224177, 2020.

[75] L. McCarthy, B. Walters, A. Amirsoleimani, and M. R. Azghadi, "Enhancing neuromorphic pattern selectivity for imbalanced data by adapting from homeostasis," in *IEEE International Symposium on Circuits and Systems*, pp. 3161–3165, 2026.

[76] A. Jeyasothy, S. Ramasamy, and S. Sundaram, "Meta-neuron learning based spiking neural classifier with time-varying weight model for credit scoring problem," *Expert Systems with Applications*, vol. 178, p. 114985, 2021.

[77] A. Mishra, S. Kumar, A. Lingamoorthy, A. Das, and N. Kandasamy, "Wafer2Spike: Spiking neural network for wafer map pattern classification," in *IEEE International Test Conference*, 2024.

[78] M. Li, X. Yang, and C. Deng, "Tackling long-tailed data challenges in spiking neural networks via heterogeneous knowledge distillation," in *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 1404–1412, 2025.

[79] O. Banjo and B. Ghoraani, "A cascaded quantized spiking neural network for real-time ECG arrhythmia detection on edge hardware," *Sensors*, vol. 26, p. 3723, 2026.

[80] D. Yu, X. Du, L. Jiang, H. Zhang, and S. Deng, "Exploiting label skewness for spiking neural networks in federated learning," in *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 6895–6903, 2025.

[81] K. Hu *et al.*, "PS-SNN: Pattern separation learning for expandable spiking neural networks in class-incremental learning," *Scientific Reports*, vol. 16, p. 12653, 2026.

[82] H. Hazan, D. Saunders, D. T. Sanghavi, H. Siegelmann, and R. Kozma, "Unsupervised learning with self-organizing spiking neural networks," in *International Joint Conference on Neural Networks*, pp. 1–6, 2018.

[83] L. R. Weisdorf *et al.*, "Ultra-long-term subcutaneous home monitoring of epilepsy—490 days of EEG from nine patients," *Epilepsia*, vol. 60, pp. 2204–2214, 2019.

[84] S. Z. Diederichsen *et al.*, "Natural history of subclinical atrial fibrillation detected by implanted loop recorders," *Journal of the American College of Cardiology*, 2019.

[85] J. Wu, Y. Zhan, Z. Peng, X. Ji, G. Yu, R. Zhao, and C. Wang, "Efficient design of spiking neural network with STDP learning based on fast CORDIC," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 6, pp. 2522–2534, 2021.

[86] W. Guo, H. E. Yantir, M. E. Fouda, A. M. Eltawil, and K. N. Salama, "Toward the optimal design and FPGA implementation of spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 8, pp. 3988–4002, 2022.

[87] A. Carpegna, A. Savino, and S. Di Carlo, "Spiker: an FPGA-optimized hardware accelerator for spiking neural networks," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 14–19, 2022.

[88] Z. He, C. Shi, T. Wang, Y. Wang, M. Tian, X. Zhou, P. Li, L. Liu, N. Wu, and G. Luo, "A low-cost FPGA implementation of spiking extreme learning machine with on-chip reward-modulated STDP learning," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1657–1661, 2022.

[89] B. Vallejo-Mancero, J. Madrenas, and M. Zapata, "Real-time execution of SNN models with synaptic plasticity for handwritten digit recognition on SIMD hardware," *Frontiers in Neuroscience*, vol. 18, p. 1425861, 2024.

[90] Q. Wang, Y. Li, B. Shao, S. Dey, and P. Li, "Energy efficient parallel neuromorphic architectures with approximate arithmetic on FPGA," *Neurocomputing*, vol. 221, pp. 146–158, 2017.

[91] T. Yousefi, G. Zoidl, and H. Kassiri, "Closed-loop 100-channel highly-scalable retinal implant with 1.02-μW analog ED-based adaptive-threshold spike detection and Poisson-coded temporally distributed optogenetic stimulation," in *2024 IEEE International Solid-State Circuits Conference (ISSCC)*, 2024.