ISLAMIC UNIVERSITY OF TECHNOLOGY

A subsidiary organ of Organization of Islamic Cooperation (OIC)

BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

From Lightweight CNNs to SpikeNets: Benchmarking Accuracy–Energy Tradeoffs with Pruned Spiking SqueezeNet

Tawsif Tashwar Dipto
200041105
Radib Bin Kabir
200041122
Mehedi Ahamed
200041125

Department of Computer Science and Engineering
Islamic University of Technology
September, 2025

From Lightweight CNNs to SpikeNets: Benchmarking Accuracy–Energy Tradeoffs with Pruned Spiking SqueezeNet

Tawsif Tashwar Dipto
200041105
Radib Bin Kabir
200041122
Mehedi Ahamed
200041125

Department of Computer Science and Engineering
Islamic University of Technology
September, 2025

Declaration of Candidate

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by **Tawsif Tashwar Dipto**, **Radib Bin Kabir**, and **Mehedi Ahamed** under the supervision of **Dr. Md. Hasanul Kabir**, Professor, Department of Computer Science and Engineering and co-supervision of **Sabbir Ahmed**, Assistant Professor, Department of Computer Science and Engineering, Islamic University of Technology, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

Dr. Md. Hasanul Kabir
Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: September 29, 2025

Sabbir Ahmed
Assistant Professor
Department of Computer Science and Engineering
Islamic University of Technology (IUT)
Date: September 29, 2025

Tawsif Tashwar Dipto
Student ID: 200041105
Date: September 29, 2025

Radib Bin Kabir
Student ID: 200041122
Date: September 29, 2025

Mehedi Ahamed
Student ID: 200041125
Date: September 29, 2025

Dedicated to Our families, for their endless patience and encouragement throughout this journey. And to our supervisor, who challenged us, guided us, and made sure this thesis was completed though not without a few sleepless nights.

Contents

List of Figures

List of Tables

List of Abbreviations

ANN	Artificial Neural Network
SNN	Spiking Neural Network
CNN	Convolutional Neural Network
DVS	Dynamic Vision Sensor
AC	Accumulate Operation
MAC	Multiply–Accumulate Operation
BPTT	Backpropagation Through Time
FLOPs	Floating Point Operations
GPU	Graphics Processing Unit
IF	Integrate-and-Fire (Neuron Model)
LIF	Leaky Integrate-and-Fire (Neuron Model)
ReLU	Rectified Linear Unit
Grad-CAM	Gradient-weighted Class Activation Mapping
SGD	Stochastic Gradient Descent

Acknowledgement

I am profoundly grateful to my supervisor, Dr. Md. Hasanul Kabir, for his endless patience, insightful critiques, and for always knowing just when to say, “Have you tried turning it off and on again?” His expertise and encouragement were instrumental in the completion of this thesis.

A special thanks to my co-supervisor, Sabbir Ahmed, who provided invaluable support and the occasional pep talk disguised as a lecture on time management. Her unwavering belief in my work kept me going, even when the only thing I believed in was the power of procrastination.

Finally, to my parents, whose love and support have been a constant source of strength. Thank you for always being there, for pretending to understand my research when I rambled on, and for not asking too many questions about why I was still not employed after all these years.

To all of you, I extend my heartfelt appreciation and a promise: next time, I will take fewer detours through the land of streaming services.

Abstract

Spiking Neural Networks (SNNs) are emerging as energy-efficient alternatives to Convolutional Neural Networks (CNNs) for edge-device deployment. However, SNNs typically exhibit a performance gap compared to CNNs, and training them remains challenging due to the non-differentiability of spiking neurons. In this work, we systematically analyze lightweight CNN-to-SNN conversion pipelines enhanced with parameters optimization and pruning. We benchmarked models on CIFAR-10, CIFAR-100, and TinyImageNet, evaluating accuracy, F1-score, parameter count, computational complexity, and energy consumption. CNNs are profiled using MAC operations, while SNNs are measured with Accumulate (AC) and MAC operations to capture event-driven sparsity. Our results show that SNNs achieve up to **15.7×** higher energy efficiency compared to CNN baselines while maintaining comparable accuracy. Notably, pruning the SNN SqueezeNet improved CIFAR-10 accuracy by **6%** and reduced parameters by **19%** compared to the original SNN SqueezeNet. Moreover, the pruned SNN SqueezeNet-P exhibits only a **1%** lower accuracy than CNN SqueezeNet while achieving an **88.1%** improvement in energy efficiency. These findings highlight the potential of SNNs as low-power alternatives for edge intelligence and provide a benchmarked evaluation framework for future studies on energy-efficient neural networks.

Chapter 1

Introduction

Deep convolutional neural networks have revolutionized visual recognition, yet their deployment on edge and embedded platforms is constrained by high energy consumption and computational demand. To address this, lightweight CNNs, such as ShuffleNet [46], MixNet [36], MnasNet [37], and SqueezeNet [14] offer competitive accuracy with significantly fewer parameters and multiply–accumulate operations, making them more suitable for resource-constrained settings [14], [36], [46]. Concurrently, Spiking Neural Networks (SNNs) have emerged as an energy-efficient alternative, event-driven communication to dramatically reduce energy consumption during inference [27], [40].Neuromorphic hardware, exemplified by chips like Innatera's Pulsar, further illustrates the real-world potential of SNNs to enable always-on sensing with ultra-low power and latency in devices such as smart wearables and IoT sensors [7].

Despite the promise of this approach, SNNs still grapple with challenges in achieving parity in accuracy with their CNN counterparts, especially when operating under tight resource budgets. As illustrated in Fig. 1.1, conventional CNNs typically occupy the high-accuracy, high-energy region of the design space, while most SNNs fall into the low-energy, low-accuracy quadrant. This accuracy–energy gap represents a fundamental challenge in neuromorphic computing. Conversion-based methodologies, which transplant pretrained CNN weights into SNN architectures, provide a pragmatic pathway for bridging this gap, allowing modern training techniques to be leveraged while inheriting the energy efficiency of spiking computation [38]. However, the current literature lacks a systematic evaluation of compact CNN-to-SNN conversions, particularly in terms of balancing accuracy, model compactness, and energy use.

In this work, we systematically study spiking versions of lightweight CNNs including

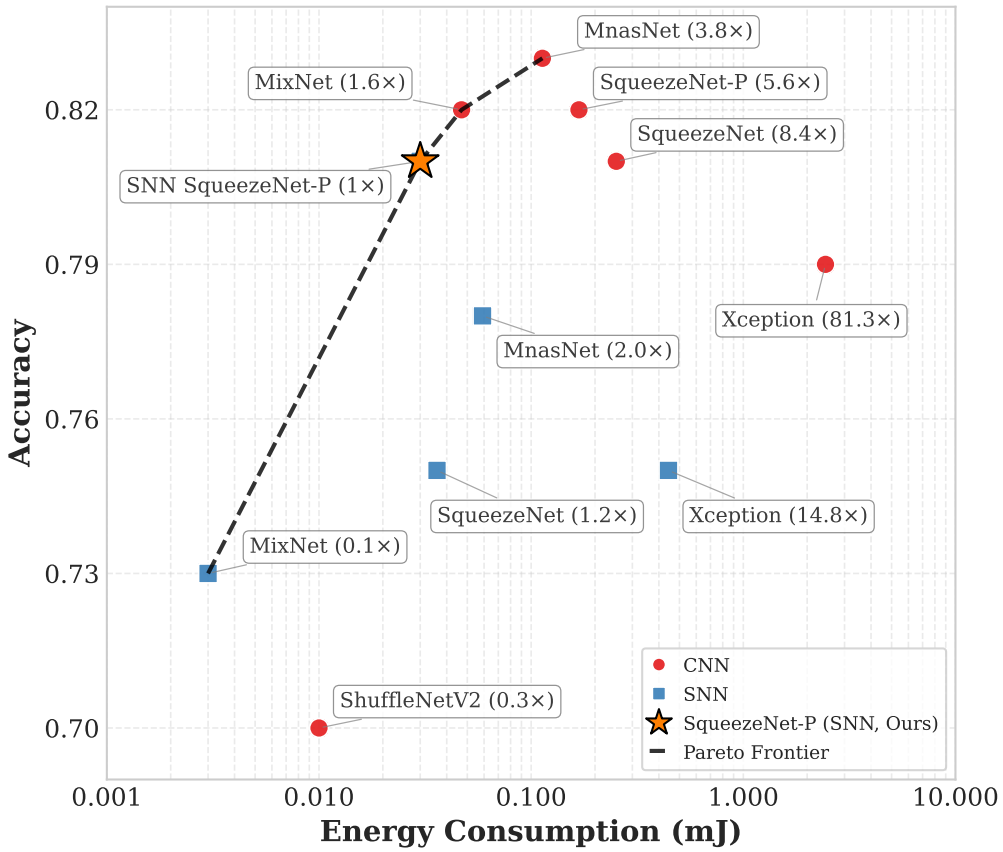


Figure 1.1: Accuracy–energy trade-off on CIFAR-10. While CNNs achieve strong accuracy, they incur much higher energy costs. SNNs reduce energy but often sacrifice accuracy. Our pruned SNN SqueezeNet-P (orange star) lies on the Pareto frontier, delivering the best balance between accuracy and efficiency.

ShuffleNet, MixNet, and SqueezeNet under consistent training and evaluation conditions. Central to our contribution is a tailored pruning strategy applied to the Spiking SqueezeNet architecture, designed to eliminate redundant pathways and further optimize efficiency. Through extensive experiments on CIFAR-10, CIFAR-100, and Tiny ImageNet, we assess models not only on traditional accuracy and F1 metrics but also on compute metrics (ACs/MACs), parameter count, and energy consumption estimates grounded in realistic hardware models. Our findings show that carefully optimized spiking networks can capture most of the performance of lightweight CNNs while significantly reducing computational load and energy requirements. As demonstrated by the Pareto-optimal position of our SNN SqueezeNet-P in Fig. 1.1, our approach achieves the best balance between accuracy and energy efficiency, paving the way for practical, energy-efficient deep learning on edge platforms. This study offers both empirical insights and practical design guidelines for deploying compact and high-performing SNNs in real-world, power-constrained environments. This introductory chapter is structured to provide both conceptual background and research

context. It begins by presenting overview on deep learning, CNNs, and the emergence of SNNs as energy-efficient alternatives. This is followed by a discussion of the motivations and scope of this study, after which we clearly state the research problem being addressed. Next, the main challenges encountered in this line of research are elaborated. Finally, the specific contributions of this work are presented and the overall organization of the thesis is outlined.

1.1 Overview

Convolutional neural networks (CNNs) have become the cornerstone of modern machine learning research, especially in computer vision. Since the breakthrough of AlexNet [2], [16] in 2012, CNNs have set the standard for tasks such as image classification, object detection, semantic segmentation, and even areas beyond vision, including speech processing and reinforcement learning. Their strength lies in the ability to automatically extract hierarchical representations of data through convolutional layers, progressively capturing low-level features such as edges and textures before moving on to higher-level semantic concepts. This powerful representational capacity has enabled CNNs [34] to achieve state-of-the-art performance across a wide range of benchmarks, often surpassing human-level accuracy in constrained visual recognition tasks [10]. As a result, CNNs have been widely adopted in academic research, industrial applications, and commercial systems alike.

Despite these achievements, the applicability of CNNs to resource-constrained environments such as embedded platforms, mobile devices, and Internet-of-Things (IoT) sensors remains problematic [21]. A key issue lies in their extremely high computational and energy demands. For example, conventional architectures such as VGG-16 [35] or ResNet-152 [10] require billions of multiply–accumulate (MAC) operations for a single forward pass, alongside hundreds of megabytes of parameter storage. Such requirements are feasible in datacenter environments equipped with high-performance GPUs, but they are unsustainable in scenarios where computational resources, power budgets, and memory availability are strictly limited. This challenge is particularly acute in edge computing, where devices must often operate in real time, powered by batteries, and without continuous access to cloud resources. The energy overhead of running conventional CNNs under these conditions not only reduces efficiency but also limits the scope of applications where deep learning can be practically deployed.

To address these issues, researchers have proposed a variety of lightweight CNN architectures that aim to reduce both the number of parameters and the overall com-

putational burden while maintaining competitive levels of accuracy. Examples include ShuffleNet [45], MixNet [44], and SqueezeNet [13], each of which introduces architectural innovations that improve efficiency. ShuffleNet leverages channel shuffling and group convolutions to reduce redundancy in feature maps, MixNet explores a systematic combination of depthwise convolutions with varying kernel sizes, and SqueezeNet achieves AlexNet-level accuracy with dramatically fewer parameters by employing "fire modules" that squeeze and expand channels in a balanced manner. These architectures demonstrate that careful architectural design can significantly reduce computational requirements while preserving accuracy, thereby making CNNs more suitable for mobile and embedded platforms. Nevertheless, even the most efficient lightweight CNNs fundamentally rely on dense computation, which inherently limits the extent of energy savings achievable through architectural modifications alone.

In parallel with these advances, spiking neural networks (SNNs) have emerged as a radically different paradigm inspired by the event-driven communication mechanisms observed in biological neurons. Unlike CNNs, which rely on continuous-valued activations and synchronous updates, SNNs transmit information through discrete spike events. A neuron in an SNN integrates input currents over time and emits a spike only when its membrane potential crosses a certain threshold. This event-driven computation has several important implications. First, it naturally introduces sparsity into the network's activity, as neurons remain inactive most of the time. Second, spikes are binary and temporally localized, reducing the amount of computation needed compared to dense floating-point operations. Third, the asynchronous nature of spike communication is highly amenable to specialized neuromorphic hardware platforms, which are designed to exploit this sparsity for dramatic reductions in power consumption [18].

The shift from continuous to event-driven computation provides significant potential for energy savings, especially when inference is performed in scenarios with sparse input signals, such as low-activity sensory streams or environments where only a small subset of neurons are required to fire at any given time. Neuromorphic hardware platforms, including Intel's Loihi, IBM's TrueNorth, and Innatera's Pulsar, exemplify the promise of SNNs by offering dedicated architectures that implement spiking neurons and synapses in a manner that directly exploits event sparsity. These systems have demonstrated orders-of-magnitude improvements in energy efficiency compared to conventional digital processors running CNNs. For example, they enable always-on sensing applications where ultra-low power consumption is critical, such as wearable devices, biomedical implants, and distributed IoT sensor networks. The ability

to maintain continuous operation under tight power budgets opens new opportunities for embedding intelligence in environments where traditional CNNs would be impractical.

While these advantages are compelling, it is important to acknowledge that SNNs are still in the early stages of practical adoption. Their representational capacity and training methodologies are not yet as mature as those of CNNs. Unlike conventional neural networks that are trained via backpropagation, SNNs must deal with the non-differentiability of spike events, which complicates gradient-based optimization [25]. Conversion-based approaches, where pretrained CNN weights are mapped to SNN architectures, provide one pathway for leveraging the strengths of CNN training while inheriting the efficiency of spiking computation. However, such conversions often suffer from accuracy degradation, especially on more complex datasets, and the field lacks systematic evaluations of how lightweight CNN-to-SNN conversions perform in terms of both accuracy and efficiency. These challenges highlight the importance of developing principled methods that close the gap between CNNs and SNNs, thereby unlocking the full potential of spiking networks as energy-efficient alternatives for edge intelligence.

1.2 Motivations and Scope

The motivation for this work arises from the rapidly expanding demand for intelligent edge devices capable of performing complex data processing locally while operating under stringent energy constraints. Over the past decade, the number of Internet-of-Things (IoT) devices [41] has grown into the billions, spanning applications from smart homes and wearable technologies to autonomous vehicles, surveillance systems, and biomedical implants. These devices are often expected to perform continuous sensing, analysis, and decision-making in real time, frequently under conditions where access to cloud computing resources is limited or unreliable. Transmitting raw data to cloud servers for processing introduces not only significant latency but also severe privacy concerns, as sensitive information such as health records, audio streams, or video feeds may be exposed during transmission. Consequently, there is an urgent need for computational intelligence that is executed directly on edge devices, ensuring both efficiency and data sovereignty.

Traditional CNNs, while highly effective in achieving high recognition accuracy, are fundamentally unsuited to these environments due to their prohibitive energy demands. Running CNN inference continuously on mobile or embedded processors

leads to rapid battery depletion, excessive heat generation, and degraded user experience. For large-scale IoT deployments, the cumulative energy footprint of conventional deep learning could become environmentally and economically unsustainable. Even lightweight CNN architectures, though considerably more efficient than their larger counterparts, still rely on dense floating-point computation that scales poorly when multiplied across millions of devices [42]. As a result, CNNs, despite their undeniable success in research and cloud-scale applications, cannot be considered a long-term solution for the pervasive deployment of artificial intelligence at the edge.

Spiking Neural Networks (SNNs) offer a fundamentally different computational model that promises to overcome these limitations. By mimicking the sparse and event-driven communication mechanisms of the brain, SNNs drastically reduce the number of operations required during inference. Neurons in an SNN remain inactive until sufficient input arrives to trigger a spike, which means that power consumption scales with the actual information content of the signal rather than the dimensionality of the input. In sensory scenarios such as always-on audio detection, low-frame-rate video monitoring, or biomedical signal processing, this event-driven paradigm allows the network to consume energy only when meaningful information is present. This property positions SNNs as transformative candidates for sustainable edge intelligence, capable of delivering high responsiveness at orders-of-magnitude lower energy budgets compared to CNNs.

The scope of this research is therefore to rigorously investigate whether SNNs can bridge the gap between theoretical promise and practical deployment. Specifically, this work focuses on lightweight CNN-to-SNN conversion pipelines, which constitute a pragmatic approach for bringing the accuracy of modern CNN training techniques into the spiking domain. Unlike from-scratch training methods, conversion leverages pretrained CNN models and adapts them into spiking architectures while maintaining representational power. However, naive conversion often results in accuracy degradation, motivating the need for enhanced strategies such as pruning, normalization adjustments, and gradient-aware optimization. By systematically evaluating these strategies, this research aims to identify designs that maximize both efficiency and accuracy.

The experimental scope of this thesis includes widely recognized benchmarks such as CIFAR-10, CIFAR-100, and Tiny ImageNet, which collectively provide a spectrum of difficulty levels and dataset scales. Evaluations are conducted not only in terms of accuracy and F1-score but also through detailed measurements of parameter counts, multiply–accumulate (MAC) operations, accumulate (AC) operations, and energy con-

sumption estimates. These multiple dimensions of evaluation are critical for understanding the practical trade-offs between accuracy, model compactness, and energy efficiency. Moreover, by grounding energy estimates in realistic neuromorphic hardware models, the study ensures that the findings are directly relevant to deployment scenarios rather than remaining purely theoretical.

Ultimately, the motivation for this research extends beyond academic curiosity. It reflects a broader technological and societal push toward energy-conscious computing. As AI becomes an integral part of everyday devices, sustainability considerations demand solutions that minimize power usage [39] without sacrificing performance. By establishing systematic methodologies and empirical benchmarks for CNN-to-SNN conversions, this work seeks to contribute not only to the advancement of neuromorphic computing research but also to the practical realization of intelligent, low-power edge devices that can scale sustainably to billions of units.

1.3 Problem Statement

Despite progress in neuromorphic computing, three core problems persist. First, there remains a notable accuracy gap between CNNs and their spiking counterparts. Second, training SNNs is inherently difficult due to the non-differentiability of spiking neurons, which limits the effectiveness of backpropagation-based learning. Third, the field lacks systematic evaluations of compact CNN-to-SNN conversions, making it challenging to identify designs that balance compactness, accuracy, and efficiency. Addressing these problems forms the central goal of this research.

To this end, we convert lightweight CNN architectures into their SNN counterparts and conduct a comparative analysis across models. Among these, SqueezeNet emerged as the most promising candidate. We further optimized the converted SqueezeNet-SNN through hyperparameter tuning and structural pruning, guided by an ablation study, to achieve an optimal balance between performance and efficiency.

1.4 Research Challenges

Developing Spiking Neural Networks (SNNs) that achieve both high accuracy and strong energy efficiency is far from straightforward, as the design process is constrained by several interrelated technical challenges. The first and perhaps most fundamental difficulty stems from the non-differentiable nature of spike generation. In biological neurons and their spiking counterparts, outputs are binary events spikes that oc-

cur when the membrane potential exceeds a threshold. This discontinuity prevents the direct application of backpropagation, the cornerstone of modern deep learning. To address this, researchers have proposed surrogate gradient techniques, which approximate the derivative of the spiking function using continuous relaxations. While such approaches enable gradient-based optimization, they introduce bias and approximation errors, limiting the ultimate accuracy attainable. Conversion-based methods, which transfer weights from pretrained artificial neural networks (ANNs) into SNN architectures, provide an alternative, but they often lead to information loss due to mismatches in temporal dynamics, membrane potential accumulation, and thresholding behavior. These training-related challenges represent a critical barrier to closing the performance gap between CNNs and SNNs.

A second major hurdle is model compactness versus representational power. The promise of SNNs lies in their potential to enable lightweight, low-power deployment, but this requires pruning and compression to eliminate redundancies. However, in already compact architectures such as SqueezeNet, aggressive pruning risks damaging critical pathways that are essential for preserving feature extraction capacity. Unlike large CNNs, which contain significant redundancy that can be safely removed, lightweight SNNs operate closer to the limits of representational sufficiency. Striking the right balance between reducing parameters for efficiency and retaining accuracy for usability remains an open problem. This challenge is compounded when deploying models across diverse datasets of varying complexity, as pruning strategies that succeed on simple datasets like CIFAR-10 may fail to generalize to more challenging benchmarks such as Tiny ImageNet.

Another layer of complexity arises from the gap between software-level simulations and hardware-level deployment. Most SNN experiments today are conducted in simulation environments where event-driven sparsity is modeled in terms of accumulate (AC) and multiply–accumulate (MAC) operations. However, neuromorphic hardware platforms such as Intel Loihi, IBM TrueNorth, or Innatera's Pulsar exhibit unique characteristics that cannot be fully captured in simulation. Factors such as spike routing overhead, memory access patterns, on-chip buffering, and communication latency between cores can significantly alter the actual efficiency observed in practice. A model that appears energy-efficient in simulation may suffer from bottlenecks when deployed on hardware due to issues like load imbalance, limited memory bandwidth, or routing congestion. Bridging this gap between theoretical efficiency and practical deployability is a central challenge in making SNNs viable for real-world applications.

Evaluation itself also poses challenges. Conventional deep learning metrics such as

FLOPs or MAC counts fail to adequately describe the computation in spiking networks, where the event-driven nature of communication alters the cost model. For SNNs, the number of accumulate operations (ACs), spike rates, and sparsity patterns must be considered alongside MACs to provide a fair assessment of energy consumption. Moreover, accuracy alone cannot capture the full picture of model quality. A highly accurate SNN that consumes disproportionate energy is unsuitable for edge devices, while an extremely energy-efficient SNN with low accuracy is equally impractical. This necessitates multi-dimensional evaluation frameworks that consider accuracy, F1-score, model size, latency, and energy in tandem. Designing and standardizing such metrics is a non-trivial challenge that the field has yet to fully resolve.

Finally, there is the broader issue of scalability and generalization. Most current SNN research focuses on relatively small-scale datasets such as MNIST or CIFAR-10, which, while useful for proof-of-concept, do not adequately represent the demands of real-world vision applications. Scaling SNNs to handle more complex datasets without losing their energy advantages is a non-trivial research question. Moreover, the interaction of different optimization strategies conversion, surrogate gradient training, pruning, normalization, and hardware mapping creates a vast design space with many unexplored trade-offs.

In summary, the development of SNNs that are both accurate and energy-efficient requires addressing challenges that span theoretical training limitations, architectural constraints, pruning trade-offs, hardware deployment considerations, and evaluation methodologies. These difficulties highlight the need for comprehensive frameworks that integrate design with hardware-aware optimization and robust benchmarking, ensuring that SNNs can move from promising laboratory prototypes to reliable components of next-generation edge intelligence.

1.5 Contributions

This thesis addresses the challenges of optimizing lightweight spiking neural networks (SNNs) for efficient image classification by bridging architectural, algorithmic, and benchmarking gaps. The key contributions are summarized as follows:

- **Development of Lightweight SNN Architectures**: Converted and adapted multiple lightweight convolutional neural networks (CNNs)—including ShuffleNet, MnasNet, MixNet, and SqueezeNet—into their spiking counterparts, enabling systematic exploration of efficiency-oriented designs in the SNN domain.

- **Gradient-Aware Learning Enhancements**: Integrated gradient-aware loss functions (e.g., Grad-CAM loss) into SNN training pipelines to improve feature discrimination and enhance gradient flow, thereby mitigating training instabilities commonly observed in deep SNNs.
- **Structural Pruning for SNN-SqueezeNet**: Proposed a tailored structural pruning strategy specifically designed for SNN-SqueezeNet. Results show that selective pruning not only reduces model complexity and energy usage but also improves classification accuracy by alleviating vanishing gradient effects.
- **Comprehensive Benchmarking**: Conducted the first broad benchmarking study of lightweight SNNs across multiple standard image classification datasets, providing systematic evaluations of accuracy, energy efficiency, and architectural behavior.
- **CNN–SNN Comparative Analysis**: Offered an in-depth comparison between CNNs and their SNN counterparts, highlighting the accuracy–energy trade-off and demonstrating the complementary strengths of both paradigms for deployment on resource-constrained hardware.

Together, these contributions advance the understanding and practical deployment of lightweight SNNs, bridging critical gaps between architecture design, training methodology, and empirical evaluation. By systematically exploring the conversion of efficient CNNs into spiking counterparts, enhancing learning through gradient-aware strategies, and optimizing model complexity via structural pruning, this work provides both theoretical insights and practical guidelines for energy-efficient image classification. The benchmarking and comparative analyses further offer a reference point for future research, enabling informed decisions when selecting or designing SNNs for resource-constrained applications. Collectively, this thesis lays a foundation for more effective, scalable, and deployable spiking neural networks.

Chapter 2

Related Works

The development of compact and energy-efficient neural networks has been the subject of extensive research across both the machine learning and neuromorphic computing communities. We begin by tracing the evolution of ANN-to-SNN conversion methods, followed by an examination of lightweight CNN architectures. We then turn to the literature on pruning techniques in neural networks, which plays a critical role in our proposed optimization framework. Finally, we review the benchmarking of SNNs across standard datasets and hardware platforms, setting the stage for our own evaluation.

2.1 ANN-to-SNN Conversion Methods

The conversion of artificial neural networks (ANNs) to spiking neural networks (SNNs) has emerged as a pragmatic solution for reducing the energy consumption of conventional deep learning models while exploiting the sparse, event-driven computation inherent to spiking architectures. This approach was motivated by the observation that training SNNs directly using biologically inspired learning rules such as spike-timing-dependent plasticity (STDP) often led to inferior accuracy compared to backpropagation-trained ANNs. By leveraging pre-trained CNNs, researchers sought to bypass the difficulties of training SNNs from scratch while preserving much of the discriminative power of conventional models.

One of the earliest systematic conversion methods was introduced by Diehl et al. [5], who proposed replacing ReLU activations with integrate-and-fire neurons. This method relied on rate coding, where input intensities were translated into spike frequencies, and careful threshold balancing ensured that the firing rates of neurons matched the

activations of their ANN counterparts. Although this approach achieved competitive results on MNIST, its scalability to more complex datasets such as CIFAR-10 was limited due to the high number of simulation time steps required. Nevertheless, the work was pivotal in establishing the feasibility of ANN-to-SNN conversion and opened the door to more sophisticated techniques.

A major challenge in conversion lies in handling deeper architectures with normalization layers and non-linear activations. Rueckauer et al. [29] addressed this by proposing methods to incorporate batch normalization and biases into SNNs, thereby extending conversion applicability to modern CNN architectures. Their approach achieved strong performance, with accuracies of 99.1% on MNIST and 91.6% on CIFAR-10. However, the reliance on thousands of simulation time steps limited practical deployment, highlighting the fundamental tension between accuracy preservation and inference latency in converted networks.

Sengupta et al. [30] sought to mitigate this latency issue by introducing signed neurons with imbalanced thresholds, which enabled VGG networks to achieve 91.55% accuracy on CIFAR-10 with only 250 time steps. This represented a significant step forward, as it demonstrated that accuracy could be preserved with far fewer spikes, making SNN inference more practical. Yet, the method also revealed the difficulty of generalizing conversion techniques across architectures: while effective for VGG-style models, its efficacy on lightweight CNNs was less clear.

More recent innovations have further reduced inference costs. Han et al. [8] proposed the residual membrane potential (RMP) technique, which allows spiking neurons to carry information across time steps without full resets. This modification accelerated convergence and enabled CIFAR-10 accuracy of 93.63% using just 32 time steps a dramatic reduction compared to earlier methods. Similarly, Ngu and Lee [26] introduced layerwise and channelwise threshold balancing, providing finer control over spike rates and reducing conversion losses across datasets including MNIST, Fashion-MNIST, and CIFAR-10. These methods illustrate the ongoing trajectory toward low-latency, high-accuracy conversion pipelines.

Figure 2.1 illustrates the fundamental differences between ANN and SNN neuron-level processing. While ANNs operate on continuous values derived directly from pixel intensities, SNNs encode information into discrete spikes, which propagate through the network only when thresholds are crossed. This difference has profound implications for energy efficiency, since inactive neurons consume negligible power. However, it also complicates the design of conversion pipelines, as the dynamics of spiking neurons are not trivially mapped from continuous activations.

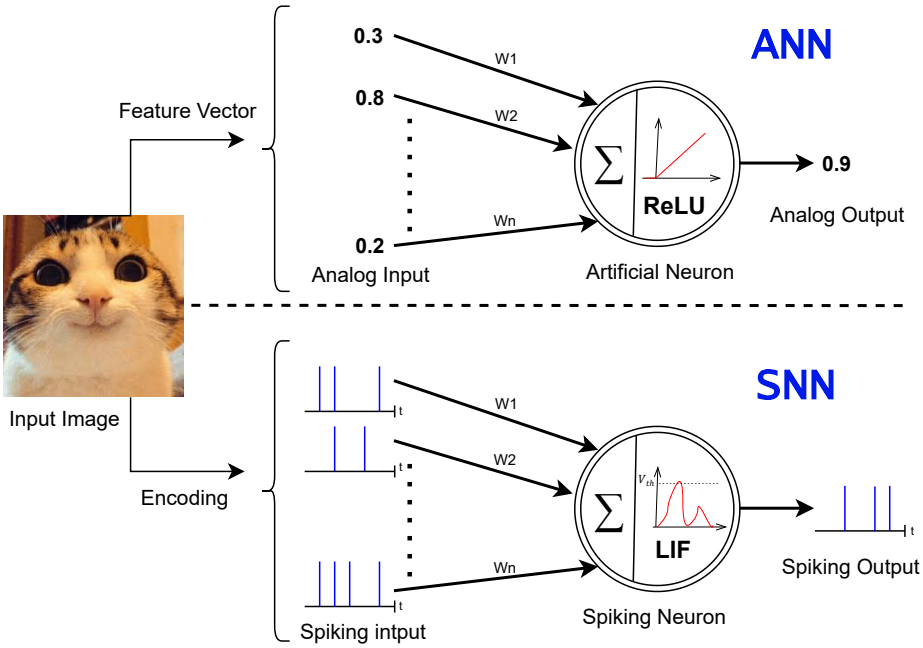


Figure 2.1: Comparison of neuron-level processing in ANN and SNN[1].

The literature on ANN-to-SNN conversion thus reflects an ongoing balancing act between three competing goals: maintaining classification accuracy, reducing the number of simulation time steps, and ensuring energy efficiency on neuromorphic hardware. Despite significant progress, challenges remain in applying these methods to lightweight CNN architectures. Much of the existing work has focused on deep and wide models such as VGG or ResNet, whereas the performance of converted compact models has been less thoroughly explored. This gap provides the foundation for the present study, which seeks to evaluate conversion methods specifically in the context of lightweight architectures like ShuffleNet, MixNet, and SqueezeNet.

2.2 Lightweight CNN Architectures

Parallel to research on SNNs, the broader deep learning community has devoted considerable effort to designing compact CNNs that reduce computational burden without drastically compromising accuracy. These efforts are particularly relevant to embedded and edge computing, where hardware constraints limit memory and power availability.

One of the earliest lightweight CNNs was SqueezeNet, illustrated in Figure 2.2, introduced by Iandola et al. [14]. The architecture employed “fire modules” that used 1x1 convolutions to reduce channel dimensionality before applying 3x3 filters, thereby

achieving AlexNet-level accuracy on ImageNet with 50x fewer parameters. This represented a paradigm shift in compact model design, demonstrating that aggressive parameter reduction was possible without catastrophic performance loss.

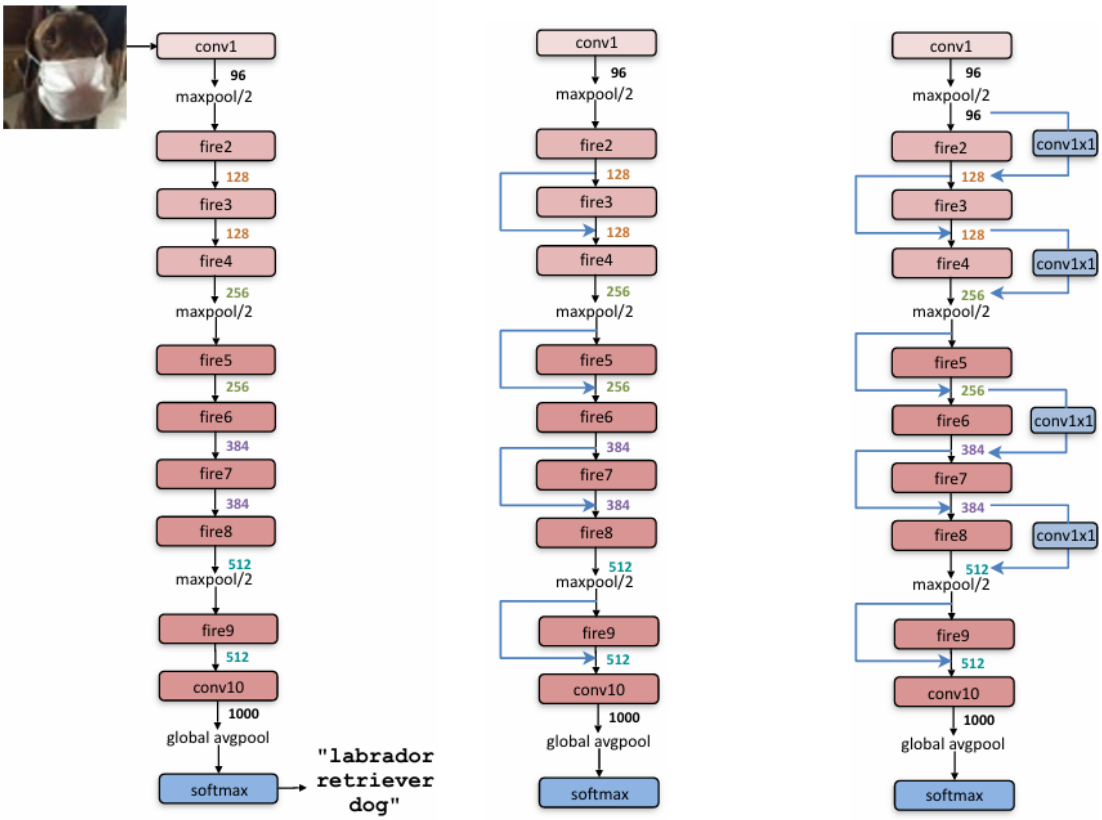


Figure 2.2: Macroarchitectural view of the SqueezeNet architecture. Left: SqueezeNet; Middle: SqueezeNet with simple bypass; Right: SqueezeNet with complex bypass. Figure adapted from [14].

Subsequent models refined this principle. ShuffleNet [46], illustrated in Figure 2.3, introduced channel shuffling in combination with depthwise separable convolutions, ensuring efficient feature mixing while keeping computational costs low. This architecture was particularly well suited for mobile devices, offering strong accuracy-cost trade-offs. MixNet [36], illustrated in Figure 2.4, extended the idea of depthwise convolutions by mixing kernels of different sizes within the same layer, enabling flexible receptive fields and achieving improved accuracy on classification benchmarks.

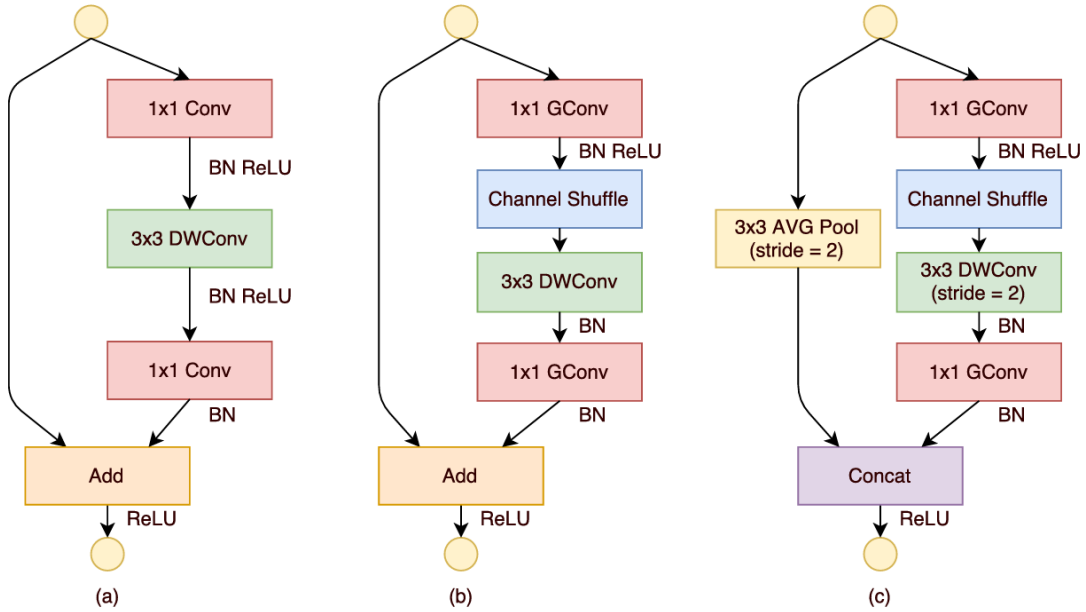


Figure 2.3: ShuffleNet Units. a) bottleneck unit with depthwise convolution; b) ShuffleNet unit with pointwise group convolution and channel shuffle; c) ShuffleNet unit with stride = 2. Figure adapted from [46].

Other noteworthy architectures include MobileNet and its successors [12], which systematically popularized depthwise separable convolutions and inverted residuals. These models pushed the state-of-the-art in mobile-friendly networks and inspired a generation of hardware-efficient designs. Despite their compactness, however, even lightweight CNNs are still based on dense multiply-accumulate (MAC) operations. When deployed on resource-constrained devices, the associated energy cost remains significant.

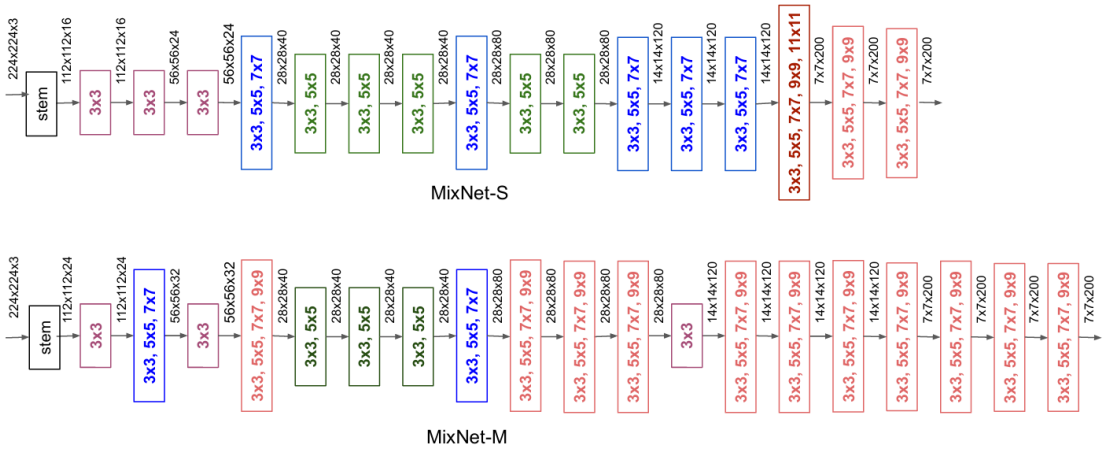


Figure 2.4: MixNet architectures– MixNet-S and MixNet-M. The mainly highlight MDConv kernel size (e.g.3x3,5x5) and input/output tensor shape. Figure adapted from [36].

For the purposes of ANN-to-SNN conversion, lightweight CNNs present both opportunities and challenges. On one hand, their small parameter count aligns naturally with the goal of efficient SNN deployment. On the other, operations such as channel shuffling and mixed kernel convolutions complicate direct conversion pipelines. The interaction between lightweight design principles and spiking computation has not been systematically studied in prior work, leaving an important gap that this thesis addresses.

2.3 Pruning Techniques in Neural Networks

Pruning refers to the process of eliminating redundant parameters or connections in neural networks, thereby reducing size, memory footprint, and computational load. This line of research has been active since the 1990s, but recent deep learning applications have renewed interest in pruning as a strategy for efficient deployment.

Han et al. [9] introduced one of the most influential frameworks, illustrated in Figure 2.5, combining magnitude-based pruning with weight quantization and Huffman coding to compress CNNs by an order of magnitude without significant accuracy loss. Their work demonstrated that modern CNNs are heavily over-parameterized, with many weights contributing little to final performance. Subsequent works refined pruning granularity, moving from weight-level pruning to structured pruning at the level of filters, channels, or even entire layers.

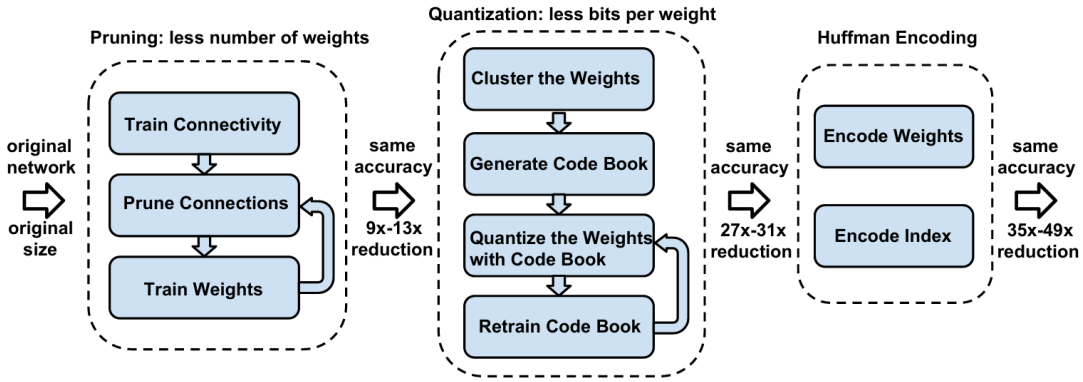


Figure 2.5: Three-stage compression pipeline consisting of pruning, quantization, and Huffman coding, achieving up to 49× compression without accuracy loss.Figure adapted from [9].

Liu et al. [20] proposed LASSO-based regularization to learn channel-level sparsity during training, enabling networks to identify redundant channels dynamically.
Molchanov et al.[22] introduced importance estimation techniques based on Taylor

expansion, further improving pruning reliability. Structured pruning has proven especially valuable for hardware deployment, since it avoids the irregular sparsity patterns that complicate memory access.

Network pruning has been extensively studied as a method to reduce model complexity while maintaining performance. Our pruning approach for SNNs builds upon established techniques in the CNN domain while considering the unique characteristics of spike-based computation.

Magnitude-based pruning, first introduced by LeCun *et al.* [17], removes connections with small weights under the assumption that they contribute minimally to network performance. Han *et al.* [9] extended this approach to deep networks, achieving 9× compression on AlexNet and 3× on VGG-16 without accuracy loss. Structured pruning methods, such as those proposed by Li *et al.* [19], remove entire filters or channels, enabling hardware acceleration and reducing inference time.

More sophisticated pruning strategies have been developed to consider the importance of network components. Molchanov *et al.* [23] proposed variational dropout for automatic relevance determination, while Yang *et al.* [43] introduced energy-aware pruning specifically for mobile devices. Liu *et al.* [20] demonstrated that pruning during training can be more effective than post-training pruning, leading to better accuracy-efficiency trade-offs.

Shen *et al.* [31] introduced ESL-SNNs, an evolutionary sparse learning method that prunes and regrows connections during training. Shi *et al.* [33] jointly pruned weights and neurons under an energy model to significantly reduce synaptic operations (SOPs). Later, Shen *et al.* [32] proposed a two-stage sparse learning scheme using the PQ index to set pruning ratios. Roy *et al.* [28] showed that SNNs inherently produce sparse activations, enabling aggressive pruning with minimal accuracy loss.

In the context of SNNs, pruning introduces additional trade-offs. While pruning reduces energy consumption by lowering spike activity and synaptic operations, overly aggressive pruning risks disrupting the delicate balance of spike propagation, leading to accuracy collapse. Recent works [3] have begun to explore pruning specifically for SNNs, but the literature remains sparse compared to the ANN domain. Moreover, most studies have focused on large architectures, leaving the combined problem of pruning lightweight CNN-to-SNN models relatively underexplored. This forms one of the optimization strategies investigated in the present study.

2.4 Evaluation of SNNs on Standard Benchmarks

Spiking Neural Networks (SNNs) have attracted increasing interest as energy-efficient alternatives to conventional deep neural networks, owing to their event-driven computation and sparse activations [1], [24], [28]. Early models relied on unsupervised STDP rules for digit recognition [5], [15], while later works introduced surrogate gradient learning, enabling deeper and more accurate architectures [6], [8], [30]. Conversion-based methods map trained CNNs into SNNs and achieve strong results on benchmarks [26], [29], but typically require high firing rates and longer inference windows. Direct training approaches, by contrast, aim to bridge the performance gap under strict latency constraints [4], [47].

In parallel, lightweight CNNs such as SqueezeNet [14], ShuffleNet [46], and MixNet [36] demonstrate how structural innovations depthwise separable convolutions, channel shuffle, or mixed kernels can drastically reduce computational cost. Coupled with pruning and compression techniques [9], [19], [43], these models offer compact yet accurate alternatives for resource-limited environments.

On standard benchmarks, computationally heavy CNNs such as ResNet and VGG routinely achieve >95% on MNIST and >75% on CIFAR-100, setting the bar for accuracy. In contrast, directly trained SNNs typically report ∼85–90% on MNIST and ∼65–70% on CIFAR-100 [4], [47], reflecting the inherent trade-off between accuracy and efficiency. Recent studies have begun combining efficiency-driven CNN principles with SNN training [27], [40], highlighting the promise of lightweight spiking architectures for neuromorphic deployment despite the remaining performance gap.

2.5 Research Gaps and Opportunities

Despite significant advances in the field of spiking neural networks (SNNs), particularly through conversion from artificial neural networks (ANNs), several key gaps remain that present opportunities for future research and innovation. While prior studies have successfully demonstrated that SNNs can approximate or even rival CNN performance on certain benchmarks, a number of limitations in methodology, scalability, and applicability continue to constrain their potential for real-world deployment. This section highlights these gaps in detail and outlines promising opportunities that emerge from addressing them.

A first major gap lies in the limited exploration of lightweight CNN-to-SNN conversions. Most existing research has focused on large and computationally heavy ar-

chitectures such as VGG and ResNet. While these networks provide a strong baseline for demonstrating accuracy preservation during conversion, they are not always practical for deployment on edge devices with strict energy and latency constraints. Lightweight architectures such as ShuffleNet, SqueezeNet, MixNet, and MnasNet remain underexplored in the SNN literature, even though they are inherently better suited for constrained environments. The opportunity here is to systematically investigate how these compact networks perform when converted to SNNs, and whether they can provide more favorable energy-accuracy trade-offs compared to their larger counterparts.

Another gap concerns the evaluation of pruning strategies in SNNs. While pruning has been extensively studied in traditional CNNs, its application in spiking models is still in its infancy. Few studies have systematically examined how pruning affects spike timing, temporal coding, and event-driven sparsity all of which are unique to SNN computation. Existing pruning strategies often focus on magnitude-based removal or structured filter pruning without accounting for the temporal dynamics of spikes. There is a clear opportunity to design pruning algorithms tailored to SNNs that exploit their intrinsic sparsity, preserve temporal fidelity, and improve both accuracy and energy efficiency.

There is also a methodological gap in how SNN performance is evaluated. Most prior works continue to rely heavily on accuracy and top-1 metrics, with only limited consideration of energy consumption, latency, or hardware feasibility. Moreover, when energy metrics are included, they are often estimated from simulations rather than measured on actual neuromorphic hardware. This creates a disconnect between research findings and their real-world implications. Future research could close this gap by developing standardized benchmarking protocols that include accumulate (AC) operations, memory access patterns, and energy-per-inference measured on emerging hardware platforms such as Intel Loihi, IBM TrueNorth, and Innatera Pulsar. This would yield more realistic insights into the deployability of SNN models.

A further gap exists in the area of training and optimization techniques. Surrogate gradient methods and ANN-to-SNN conversions remain the two dominant approaches, yet both have limitations. Surrogate gradients improve training feasibility but often face scalability issues and may not align well with event-driven computation. Conversion methods, on the other hand, can lead to information loss, particularly when threshold balancing or normalization fails to capture the nuances of spiking dynamics. Hybrid training approaches that combine the strengths of both paradigms, or novel optimization strategies that explicitly account for spike-based computation, rep-

resent opportunities for exploration.

In addition, there is a gap in dataset diversity. While SNN performance has been validated on standard benchmarks such as MNIST, CIFAR-10, and CIFAR-100, relatively little work has examined their behavior on larger-scale or domain-specific datasets such as Tiny ImageNet, ImageNet, or real-world sensor streams from edge devices. This lack of dataset variety makes it difficult to assess how well SNNs generalize beyond small-scale controlled benchmarks. Future research should investigate SNN performance under more diverse, noisy, and dynamic data conditions to better reflect deployment scenarios in IoT, robotics, and autonomous systems.

Finally, opportunities exist in bridging the simulation-to-hardware gap. Current SNN research largely depends on software frameworks and idealized assumptions about spike propagation, latency, and energy efficiency. However, actual neuromorphic hardware imposes constraints such as limited memory bandwidth, fixed precision arithmetic, and communication bottlenecks that are not always reflected in simulations. Research that integrates hardware-in-the-loop evaluation or co-designs algorithms alongside hardware can close this gap, ensuring that theoretical gains translate into practical efficiency.

Chapter 3

Proposed Methodology

In this work, we propose a systematic methodology for developing and evaluating lightweight Spiking Neural Networks (SNNs) for image classification tasks. The methodology is organized into several stages, beginning with the choice of benchmark datasets. We employ three well-established datasets. Those are- CIFAR-10, CIFAR-100 and TinyImageNet to ensure robustness across different levels of complexity. These datasets progressively increase the difficulty of classification, enabling us to assess the scalability of the proposed approach from simpler to more challenging scenarios.

The next stage involves converting conventional lightweight convolutional neural networks (CNNs) into their spiking counterparts. Specifically, ShuffleNet, SqueezeNet, MnasNet, and MixNet are systematically adapted by replacing ReLU activations with Leaky Integrate-and-Fire (LIF) neurons and encoding pixel intensities as current injections over multiple timesteps. This conversion ensures that the models retain biological plausibility while benefiting from the event-driven and energy-efficient nature of SNNs.

After conversion, different candidate architectures are evaluated in order to identify the most effective spiking model. Empirical results demonstrate that the SNN variant of SqueezeNet provides the best trade-off between accuracy, spike efficiency and parameter count. Consequently, SNN-SqueezeNet is selected as the base architecture for further optimization.

To improve SNN-SqueezeNet's computational and energy efficiency, a structured pruning strategy is applied whereby less effective block modules are systematically removed while preserving overall accuracy. A random search was performed to tune hyperparameters and achieve the best performance.

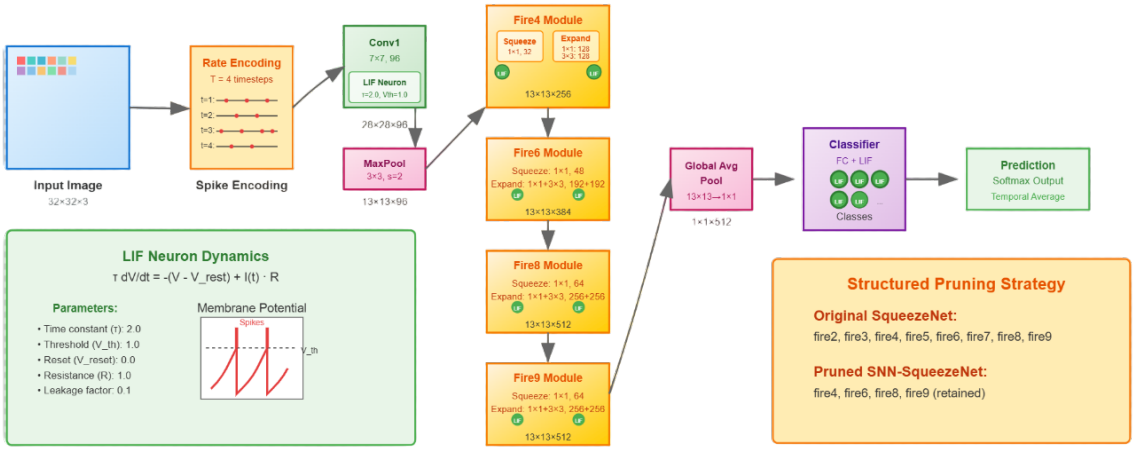


Figure 3.1: Overall architecture of the proposed pruned SNN-SqueezeNet model. The input image is first converted into spike trains using rate encoding, followed by convolution, pooling, and LIF neuron dynamics. The retained Fire modules (*Fire4*, *Fire6*, *Fire8*, and *Fire9*) are highlighted after structured pruning, reducing model complexity while preserving performance. A global average pooling layer and fully connected classifier with LIF neurons generate the final prediction.

The training strategy is designed to ensure fair and reproducible evaluation. All models are trained from scratch under controlled conditions using the Adam optimizer with a scheduled learning rate decay. To overcome the non-differentiable nature of spiking activity, surrogate gradient descent is employed while a composite loss function that combines cross-entropy with a gradient-aware regularization term is used to stabilize training and mitigate vanishing gradients.

Finally, the performance of both CNN and SNN models is assessed through a diverse set of evaluation metrics. In addition to top-1 classification accuracy and F1-score, we analyze model complexity in terms of parameter count and computational cost. Multiply-accumulate operations (MACs) are reported for CNNs, while accumulate operations (ACs) are measured for SNNs to reflect their sparse, event-driven computations. Furthermore, energy consumption is estimated using established cost models, providing a holistic perspective on the trade-offs between predictive performance and computational efficiency.

Overall, the methodology provides a controlled and comprehensive framework that balances biological fidelity, energy efficiency and practical performance, enabling a fair comparison of lightweight SNNs against their CNN counterparts.

3.1 SNN Conversion

To bridge the gap between ANNs and SNNs, we adopted a conversion-based approach starting from a non-pretrained ANN model. Specifically:

- **Neuron Replacement**: ReLU activations in the ANN were replaced with Leaky Integrate-and-Fire (LIF) neurons, a biologically plausible spiking model. The LIF neuron dynamics are governed by the following differential equation:

$$\tau \frac{dV}{dt} = -(V - V_{\text{rest}}) + I(t) \cdot R,$$

  where V is the membrane potential, τ is the time constant (set to 2.0), V_{rest} is the resting potential (0.0), $I(t)$ is the input current, and R is the resistance (1.0). A spike is emitted when V exceeds a threshold V_{th} (1.0), followed by a reset to V_{reset} (0.0). Leakage was incorporated with a factor of 0.1 to mimic realistic neuronal behavior and stabilize training.

- **Input Encoding**: Pixel intensities from the input images were directly fed into the SNN as constant current injections over a fixed number of time steps ($T = 4$), akin to rate coding. This method encodes intensity values proportionally to firing rates, where higher pixel values result in stronger inputs and more spikes. No Poisson encoding was used to maintain determinism and computational efficiency.

This conversion ensures compatibility with backpropagation while preserving the event-driven nature of SNNs, reducing energy consumption compared to traditional ANNs.

3.2 Model Architecture

To identify an optimal base architecture for our SNN, we first created SNN counterparts for several popular lightweight CNN models: ShuffleNet, XceptionNet, MNASNet, MixNet and SqueezeNet. These were converted using the aforementioned SNN conversion process and evaluated on the CIFAR-10 dataset for initial performance comparison. Empirical results showed that the SNN variant of SqueezeNet outperformed the others in terms of accuracy, spike efficiency and parameter count, making it the selected base model for further refinement.

Our base architecture is thus a spiking variant of SqueezeNet, chosen for its lightweight design and efficiency in resource-constrained environments. SqueezeNet employs fire modules that consist of a squeeze layer (1×1 convolutions) followed by expand layers

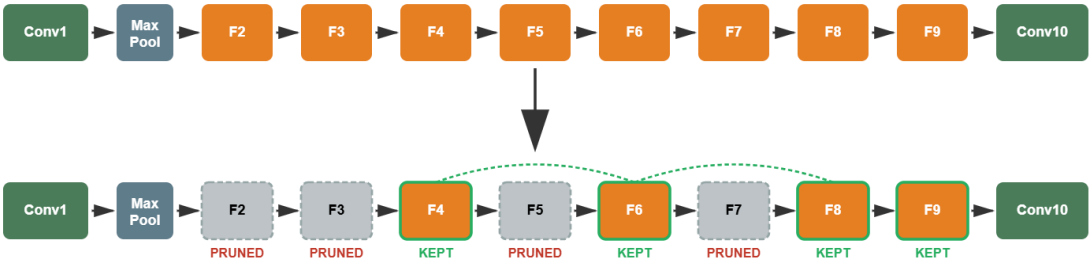


Figure 3.2: Pruning strategy for SNN SqueezeNet. Each 'Fi' denotes a Fire (firing) module 'i'. The top row shows the original SNN SqueezeNet with all Fire modules active. The bottom row illustrates the pruned SNN SqueezeNet, where ineffective Fire modules (F2, F3, F5, F7) are removed, and only selected modules (F4, F6, F8, F9) are kept. The dotted green line indicates that the retained Fire modules maintain their functional connections, ensuring information flow through the pruned network.

(mix of 1×1 and 3×3 convolutions), achieving AlexNet-level accuracy with significantly fewer parameters.

SNN SqueezeNet: We adapted SqueezeNet by integrating LIF neurons into each convolutional and fully connected layer. The network structure includes an initial convolution (conv1), followed by max-pooling, eight fire modules (fire2 to fire9), another max-pooling, and a final global average pooling leading to a softmax classifier. Spiking occurs across T time steps, with temporal accumulation of membrane potentials.

Pruned SNN-SqueezeNet: To optimize the Spiking SqueezeNet architecture for both computational efficiency and classification performance, we employed a structured pruning strategy guided by the ablation study results presented in Table 4.2. This approach systematically identifies and removes less influential layers—specifically Fire2, Fire3, Fire5, and Fire7—while retaining the modules that contribute most significantly to network performance, namely Fire4, Fire6, Fire8, and Fire9. By selectively preserving these critical modules, the pruned network achieves a substantial reduction in parameter count and computational complexity without incurring a notable drop in accuracy. The final configuration, depicted in Fig. 3.2, highlights how targeted pruning can streamline the architecture, improve energy efficiency, and maintain the representational power necessary for reliable classification. Overall, this demonstrates that strategic layer-level pruning in spiking networks can effectively balance performance and resource utilization, making it well-suited for energy-constrained and resource-limited environments.

Hyperparameter Tuning: To determine the most appropriate parameter settings, a random search strategy was employed over key hyperparameters including the learning rate, the neuronal time constant (τ), the firing threshold (V_{th}), and the type of sur-

rogate gradient used during backpropagation. The time constant τ controls the rate at which the membrane potential decays over time, thereby influencing the temporal integration of incoming spikes. The firing threshold V_{th} specifies the membrane potential value that must be exceeded for a neuron to emit a spike, effectively regulating the neuronal excitability and sparsity of spiking activity. Surrogate gradient functions, on the other hand, provide differentiable approximations to the non-differentiable spike activation function, enabling stable training of spiking neural networks via backpropagation through time (BPTT). By tuning these parameters systematically, the network is optimized for both accuracy and efficiency, ensuring reliable convergence while maintaining biologically plausible dynamics.

After random search the neuronal time constant (τ) was set to 1.08, the firing threshold (V_{th}) was set to 1.0 and learning rate was set to 0.001. Surrogate gradient function was set to ATan which gave best performance.

3.3 Experimental Setup and Training Strategy

All models, including both CNN and SNN variants, were trained from scratch for 150 epochs using the Adam optimizer with an initial learning rate of 0.001, decayed by a factor of 0.1 at epochs 50 and 100. A batch size of 8 was used due to memory constraints. Early stopping with a patience of 10 epochs was applied based on validation accuracy. To ensure a fair comparison of computational cost and energy efficiency, no pretrained weights were used for either SNN or CNN models, avoiding discrepancies in total operation count and spike activity. Spike rates were monitored for SNNs, targeting an average firing rate below 0.3 per neuron per time step. All experiments were conducted on NVIDIA Tesla P100 GPUs using PyTorch and the SpikingJelly framework.

Training SNNs presents challenges due to the non-differentiable nature of spiking functions. We addressed this using the following techniques:

- **Surrogate Gradient Descent:** Training spiking neural networks (SNNs) is challenging due to the non-differentiability of the Heaviside step function used for spike generation. To address this, surrogate gradient methods replace the non-existent derivative with a smooth, differentiable approximation during the backward pass [24]. In this work, we employ the arctangent-based surrogate

function implemented in SpikingJelly's `surrogate.ATan()`, defined as

$$\sigma'(u) = \frac{\alpha}{2} \cdot \frac{1}{1 + \left(\frac{\pi \alpha u}{2}\right)^2}, \tag{3.1}$$

where $u = v - V_{th}$ is the normalized membrane potential, v is the membrane potential, V_{th} is the firing threshold, and α is a scaling factor controlling the gradient's sharpness. This formulation yields a smooth approximation of the step function's derivative, enabling end-to-end optimization via backpropagation through time (BPTT) while retaining binary spike generation ($s = H(v - V_{th})$) in the forward pass.

It is worth noting that several alternative surrogate functions have been explored in the literature, such as the sigmoid, fast sigmoid, piecewise linear, and exponential functions [24]. While these functions differ in smoothness and computational cost, they share the same underlying principle of providing non-zero gradients in the vicinity of the firing threshold. We adopt the arctangent surrogate due to its balance between numerical stability and biological plausibility, as it produces gradients that decay gracefully away from the threshold while avoiding saturation issues often observed in exponential surrogates. Such design choices are critical for ensuring both convergence and efficient training when scaling SNNs to deeper architectures.

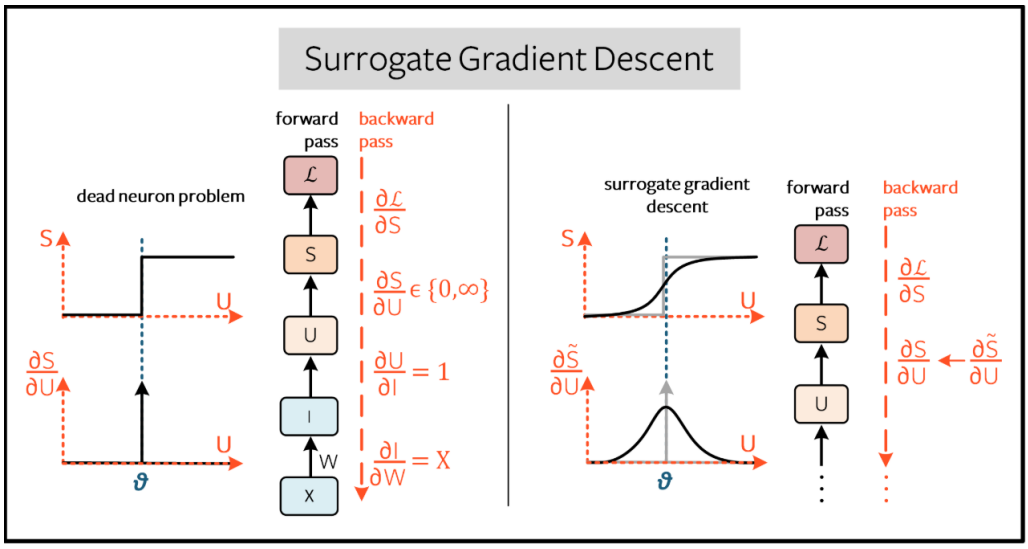


Figure 3.3: Illustration of the surrogate gradient approach. While the forward pass uses the non-differentiable Heaviside step function for spike generation, its gradient is approximated by a smooth arctangent function during the backward pass, ensuring stable and tractable learning.

- **Loss Functions:** The primary objective was multi-class classification, optimized using Cross-Entropy Loss:

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i), \tag{3.2}$$

where y_i is the ground truth and $\hat{y}_i$ is the predicted probability averaged over time steps. To mitigate vanishing gradients in deep SNNs, we introduce a Gradient-Aware Loss that penalizes low gradient magnitudes in intermediate layers:

$$\mathcal{L}_{\text{GA}} = \lambda \sum_{l} \left(1 - \frac{||\nabla_{W_l} \mathcal{L}_{\text{CE}}||_2}{||\nabla_{W_l} \mathcal{L}_{\text{CE}}||_2 + \epsilon}\right), \tag{3.3}$$

where $\nabla_{W_l} \mathcal{L}_{\text{CE}}$ is the gradient of the cross-entropy loss with respect to the weights W_l of layer l, $||\cdot||_2$ denotes the L2 norm, λ is a weighting factor, and ϵ ensures numerical stability. The total loss is:

$$\mathcal{L} = \mathcal{L}_{\text{CE}} + \mathcal{L}_{\text{GA}}. \tag{3.4}$$

This methodology provides a controlled and comprehensive framework for comparing CNN and SNN models, balancing biological fidelity, energy efficiency, and practical performance while ensuring fair and reproducible evaluation.

3.4 Evaluation Metrics

To comprehensively assess the performance of Spiking Neural Networks (SNNs) and their CNN counterparts, we employ a diverse set of evaluation metrics:

- **Accuracy (Acc)**: The top-1 classification accuracy, which measures the proportion of correctly predicted samples over the total test set.
- **F1-Score (F1)**: The harmonic mean of precision and recall, providing a balanced evaluation for datasets with potential class imbalance.
- **Parameters (Params)**: The total number of trainable parameters, reported in thousands (K) or millions (M), representing the model's memory footprint.
- **Multiply-Accumulate Operations (MACs)**: For CNNs, computational complexity is dominated by MAC operations. We estimate MACs using the `ptflops` library.

- **Accumulate Operations (ACs)**: Unique to SNNs, ACs represent spike-driven accumulation events, which incur lower energy costs due to their sparse and event-driven nature.
- **Energy Consumption**: Energy efficiency is critical for edge deployment. Following Horowitz (2014) [11], the energy cost is approximated as:

$$E_{\text{total}} = (N_{\text{AC}} \times 0.9 \text{ pJ}) + (N_{\text{MAC}} \times 4.6 \text{ pJ}), \tag{3.5}$$

  where N_{AC} and N_{MAC} denote the total number of accumulate and multiply-accumulate operations, respectively. For CNNs, only MACs are considered, while for SNNs both ACs and MACs are accounted for using the `syOps` library.

This combination of accuracy, F1-score, model complexity, and energy estimates enables a holistic evaluation of trade-offs between predictive performance and computational efficiency.

The methodology adopted in this study has a sequence of well-defined stages, each of which contributes to building an efficient and biologically plausible spiking neural network. The process begins with the careful selection of benchmark datasets of increasing complexity, including CIFAR-10, CIFAR-100, and TinyImageNet. This step ensures that the models are tested under progressively challenging conditions, thereby validating their robustness and generalizability. The datasets not only provide the basis for training and testing but also directly inform the rate encoding strategies and neuron parameterization used during the ANN-to-SNN conversion. In this way, dataset choice becomes central to both model design and performance assessment.

Second stage was the conversion of lightweight convolutional neural networks into spiking architectures. In this step, traditional activation functions such as ReLU are replaced with Leaky Integrate-and-Fire (LIF) neurons, and event-driven schemes are employed to ensure that input signals are represented in a form compatible with temporal spiking dynamics. This conversion lays the foundation for bridging conventional deep learning techniques with neuromorphic computation.

The third step of the methodology involves comparing the converted spiking architectures in order to identify the best-performing model. Lightweight CNNs such as ShuffleNet, SqueezeNet, MnasNet, and MixNet are evaluated in their spiking forms across multiple criteria, including classification accuracy, computational efficiency, and parameter count. This systematic comparison allows for the selection of an optimal baseline architecture, which in this case is SNN-SqueezeNet, as it provides the best trade-off between accuracy and efficiency.

The final stage refines the chosen architecture through structured pruning and hyperparameter tuning. Redundant block modules are removed, and the resulting leaner network is retrained with optimized hyperparameters to maximize accuracy while minimizing energy consumption and parameter usage. This process not only strengthens the performance of the best model but also ensures that the final architecture remains lightweight, efficient, and biologically plausible.

In short, the methodology can be seen as a pipeline that begins with ANN-to-SNN conversion, proceeds through dataset-driven evaluation, narrows down to the best model, and concludes with pruning and optimization. Each stage builds upon the previous one, resulting in a carefully optimized framework that balances predictive performance with computational efficiency.

Chapter 4

Results and Discussion

This chapter presents the empirical findings of the proposed study, offering a comprehensive analysis of the performance trade-offs inherent in converting lightweight Convolutional Neural Networks (CNNs) to Spiking Neural Networks (SNNs) for energy-efficient image classification. The investigation follows a structured, multi-stage approach designed to first identify a suitable base architecture and then optimize it for a superior balance between accuracy and energy consumption.

Our analysis begins with the conversion of several state-of-the-art lightweight CNNs, including **ShuffleNet [46], MnasNet [37], MixNet [36], and SqueezeNet [14]** into their spiking counterparts. These initial SNN models were systematically evaluated on benchmark datasets to establish which architecture best translates to the spiking domain in terms of accuracy, parameter count, and spike efficiency. The empirical results from this phase identified **SNN SqueezeNet** as the most promising candidate among the converted models.

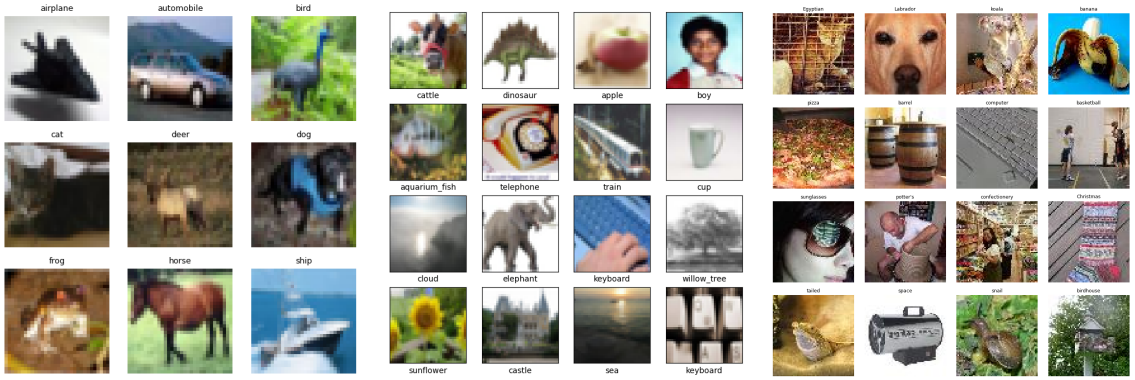
Building on this finding, the second stage of our research focused on enhancing the selected SqueezeNet architecture through a structured pruning strategy. This technique was designed not only to reduce model complexity and energy use by eliminating redundant pathways but also to improve classification accuracy.

Finally, the chapter culminates in a detailed comparative analysis, pitting the original CNNs against their SNN equivalents, including our optimized pruned model, SqueezeNet-P. This comparison critically examines the **accuracy-energy trade-off**, quantifying the substantial energy savings that SNNs can achieve with only marginal, if any, degradation in performance. The evaluation is grounded in a diverse set of metrics including accuracy, F1-score, parameter count, computational operations (MACs and ACs), and estimated energy consumption across the CIFAR-10, CIFAR-100, and

Tiny ImageNet datasets to ensure a robust and holistic assessment. The discussion will further explore ablation studies on the pruning process, analyze trends across datasets, and reflect on the broader implications of these findings for deploying practical and energy-efficient deep learning on edge platforms.

4.1 Datasets and Experimental Setup

Experiments were conducted on three benchmark image classification datasets of increasing complexity: **CIFAR-10, CIFAR-100** and **Tiny ImageNet**. CIFAR-10 contains 60,000 color images of resolution 32×32 pixels, evenly distributed across 10 classes with 6,000 images per class. The dataset is fully balanced and serves as a baseline for general-purpose vision tasks. CIFAR-100, although of the same size and resolution, is significantly more challenging as its 60,000 images are divided into 100 fine-grained classes (600 images per class), which are further grouped into 20 superclasses. The high inter-class similarity makes it a demanding benchmark. Tiny ImageNet provides an even more difficult setting, consisting of 200 categories, each with 500 training and 50 validation samples at 64×64 resolution. Unlike the CIFAR datasets, Tiny ImageNet introduces larger class diversity and complex visual backgrounds, thereby serving as a proxy for large-scale image recognition. All datasets are strictly balanced, ensuring that performance differences are attributable to architectural and training factors rather than label imbalance.



(a) Sample images from CIFAR-10. **(b)** Sample images from CIFAR-100. **(c)** Sample images from Tiny ImageNet.

Figure 4.1: Illustrative samples from the benchmark datasets used in this study. For clarity, only a subset of classes is visualized.

For preprocessing, images were resized to a fixed resolution (32×32 for CIFAR datasets and 64×64 for Tiny ImageNet), converted to tensors, and normalized channel-wise with mean $(0.5, 0.5, 0.5)$ and standard deviation $(0.5, 0.5, 0.5)$. The official training

sets were split into 80% training and 20% validation partitions using a fixed random seed for reproducibility, while the designated test sets were reserved exclusively for final evaluation. Data loading employed a batch size of 8, with shuffling enabled for training to introduce stochasticity in gradient updates, and disabled for validation and test phases to ensure consistent evaluation.

All models were implemented in PyTorch with SpikingJelly extensions to enable spiking neuron simulation. Importantly, every model-both CNN and SNN variants, was trained from scratch without relying on pretrained weights. This design choice was made to ensure a fair and consistent comparison, particularly for measuring energy consumption. Pretrained CNNs often embed optimized representational priors, which could bias the relative energy-accuracy trade-offs and distort the comparative evaluation against SNNs. Training all models from the same initialization conditions therefore provides an unbiased benchmark of energy and accuracy.

The pipeline was standardized across experiments. Training was performed on NVIDIA Tesla P100 GPUs using the Adam optimizer with an initial learning rate of 1×10^{-3}, decayed in steps during training. Models were trained for up to 150 epochs with early stopping (patience of 10 epochs) based on validation accuracy to mitigate overfitting. Temporal dynamics were modeled with $T = 4$ time steps in the SNNs to capture spatiotemporal activity while keeping simulation overhead tractable. The primary loss function was categorical cross-entropy, augmented with a Grad-CAM-aware regularization term ($\lambda_{\text{gradcam}} = 0.2$) to encourage discriminative activation patterns. Gradient approximation during backpropagation through spikes was enabled by the arctangent surrogate (ATan), which balances smooth gradient flow with preservation of spiking dynamics.

At the end of each epoch, models were validated, and both accuracy and average validation loss were monitored. The best-performing model based on validation accuracy was checkpointed and subsequently evaluated on the test set. This consistent data handling and training strategy ensures that observed differences arise primarily from architectural design choices and pruning strategies rather than implementation discrepancies.

4.2 Experimental Results

4.2.1 Performance Across Datasets

Table 4.1: Performance comparison of SNN architectures across datasets

Dataset	Network	Acc	F1	AC (k/M)	MAC (k/M)	Params (k/M)	Energy (mJ)
CIFAR10	SNN-ShuffleNetV2	0.40	0.40	**93**	**894**	911.0	0.0042
	SNN-Xception	0.75	0.75	1510	96300	1025.0	0.4444
	SNN-MnasNet	0.78	0.77	3430	12130	151.8	0.0589
	SNN-MixNet	0.73	0.73	523	506.9	**84.90**	**0.0028**
	SNN-SqueezeNet	0.75	0.74	16610	4470	736.5	0.0355
	SNN-SqueezeNet-P (ours)	**0.81**	**0.81**	11200	4230	599.1	0.0295
CIFAR100	SNN-ShuffleNetV2	0.12	0.10	**163**	894.6	1000.0	0.0043
	SNN-Xception	0.36	0.34	1780	96310	1025.0	0.4446
	SNN-MnasNet	0.49	0.48	3050	12930	163.5	0.0622
	SNN-MixNet	0.44	0.43	862	**506.9**	**108.0**	**0.0023**
	SNN-SqueezeNet	0.47	0.47	20790	4470	782.6	0.0393
	SNN-SqueezeNet-P (ours)	**0.54**	**0.53**	13490	4230	645.3	0.0316
Tiny ImageNet	SNN-ShuffleNetV2	0.11	0.09	642	3780	1110.0	0.0180
	SNN-Xception	0.23	0.21	6440	385630	10460.0	1.7797
	SNN-MnasNet	0.38	0.37	12040	51690	176.4	0.2486
	SNN-MixNet	0.34	0.33	**2000**	**203.0**	**133.7**	**0.0093**
	SNN-SqueezeNet	0.41	0.40	80094	17880	833.9	0.1543
	SNN-SqueezeNet-P (ours)	**0.45**	**0.44**	51970	16920	696.6	0.1246

Table 4.1 reports the accuracy, F1 score, AC, MAC, parameter counts, and energy consumption of six SNN architectures across CIFAR-10, CIFAR-100, and Tiny ImageNet. Overall, the pruned variant *SNN-SqueezeNet-P* consistently delivered the best performance across all three datasets, surpassing its unpruned counterpart in both accuracy and energy efficiency. In contrast, *MixNet* and *ShuffleNetV2* emerged as the most energy-efficient models, but this came at the cost of noticeably lower classification performance. On the other end of the spectrum, *Xception* demonstrated the heaviest computational and energy demands without achieving commensurate accuracy gains, underscoring the inefficiency of simply scaling up model complexity in the SNN domain. Taken together, the results highlight that pruning offers a more balanced path to improving both accuracy and efficiency, whereas extremely lightweight or overly complex models tend to underperform in one dimension or the other.

4.2.2 Ablation Study on Pruned SNN-SqueezeNet

The ablation experiments summarized in Table 4.2 were designed to understand the relative importance of different Fire modules in SNN-SqueezeNet and to identify prun-

ing schedules that achieve the best trade-off between accuracy, parameter reduction, and energy consumption. We explored several structured pruning strategies, each corresponding to the selective removal of modules from either **the head, the tail, or alternating positions** of the network. By systematically analyzing these configurations, we can assess how the depth and location of spiking computations influence both representational capacity and efficiency.

Table 4.2: Ablation study on pruning Fire modules from SNN-SqueezeNet on CIFAR-10. (✓) indicates the module is retained, (–) indicates it is pruned. Rows are grouped by pruning schedule.

Schedule	F2	F3	F4	F5	F6	F7	F8	F9	Acc.	Params	E (mJ)
Baseline											
Full Model	✓	✓	✓	✓	✓	✓	✓	✓	0.75	736,500	0.0355
Head-Heavy Pruning											
Head-1	✓	✓	✓	✓	–	–	–	✓	0.77	559,080	0.0241
Head-2	✓	✓	✓	✓	–	–	✓	–	0.77	376,220	0.0257
Tail-Heavy Pruning											
Tail-1	✓	–	–	✓	✓	✓	✓	✓	0.78	673,720	0.0304
Tail-2	✓	–	✓	–	✓	✓	✓	✓	0.79	621,660	0.0291
Alternating Pruning											
Alt-1	✓	–	✓	–	✓	–	✓	–	0.79	362,030	0.0260
Alt-2	–	✓	–	✓	–	✓	–	✓	0.76	363,050	**0.0226**
Refined Alternating											
Ref-1	–	–	✓	–	✓	–	✓	✓	**0.81**	599,130	0.0295
Ref-2	✓	–	✓	✓	✓	–	✓	✓	0.79	735,950	0.0279

The baseline model, in which all Fire modules are retained, achieves 75% accuracy on CIFAR-10 with 736.5k parameters and an energy cost of 0.0355 mJ. When pruning is concentrated toward the later stages of the network (head-heavy pruning), accuracy remains at 77%, but parameter counts drop substantially up to almost 50% relative reduction in the Head-2 configuration. However, the aggressive removal of deep modules also leads to less stable performance, suggesting that late-stage representations carry crucial discriminative power in the spiking domain. In contrast, pruning modules in the earlier part of the network (tail-heavy pruning) yields higher gains in accuracy, reaching 79% in the Tail-2 configuration, while still reducing both parameters and energy. This indicates that the early convolutional layers are somewhat redundant in the spiking representation and can be pruned without significantly compromising classification ability.

Alternating pruning schedules, in which every other module is removed, achieve a more balanced outcome. The Alt-1 configuration, which retains half of the modules in a staggered pattern, produces 79% accuracy with a parameter count reduced to 362k and an energy consumption of 0.0260 mJ. This represents one of the most efficient trade-offs observed in the study, as it compresses the model by more than 50% while simultaneously improving accuracy. The Alt-2 configuration pushes energy usage even lower to 0.0226 mJ, but its accuracy drops to 76%, showing that overly sparse representations cannot preserve performance despite their efficiency.

Finally, the refined alternating schedules demonstrate the best overall performance. In particular, the Ref-1 configuration, which retains modules F4, F6, F8, and F9, achieves the highest accuracy of 81%, a 6 percentage point improvement over the baseline. Although its parameter count remains higher than that of the alternating schedules, it still offers a meaningful reduction relative to the full model, while energy consumption decreases to 0.0295 mJ. This result highlights the significance of carefully preserving deeper modules, especially those toward the end of the architecture, which appear to capture critical spiking representations that drive generalization. The superior performance of Ref-1 also demonstrates that pruning is not merely a compression strategy but can actively enhance representational robustness by eliminating redundant pathways and focusing computation on the most informative channels.

In summary, the ablation study shows that pruning strategies strongly influence both the accuracy and energy profile of SNN-SqueezeNet. Early-stage modules contribute less critically to classification and can be removed without substantial degradation, whereas late-stage modules are essential for maintaining accuracy. The refined alternating pruning scheme achieves the most favorable balance, combining improved generalization with reduced computational cost, thereby confirming the role of structured pruning as an effective optimization strategy in spiking architectures.

4.3 Quantitative Analysis

The quantitative trends observed in Table 4.1 indicate consistent and interpretable relationships between model capacity, computational load and generalization across datasets of increasing difficulty. Most notably, the pruned variant *SNN-SqueezeNet-P* attains the best absolute accuracies on CIFAR-10, CIFAR-100 and Tiny ImageNet while simultaneously reducing parameter count and energy consumption relative to the baseline SqueezeNet. Concretely, on CIFAR-10 the pruning schedule that produced SqueezeNet-P increases accuracy from 0.75 to 0.81 (an absolute gain of 6 per-

centage points), while reducing stored parameters from 736.5k to 599.1k (a reduction of 137.4k, $\approx$ 18.7%) and lowering measured inference energy from 0.0355 mJ to 0.0295 mJ (a reduction of 0.006 mJ, $\approx$ 16.9%). The same structured pruning pattern produces similar compression on the larger datasets: on CIFAR-100 accuracy improves by 0.07 (7 percentage points, $\approx$ 14.9% relative), parameters fall by $\approx$ 137.3k ($\approx$ 17.5%) and energy drops by $\approx$ 19.6%; on Tiny ImageNet the absolute accuracy gain is 0.04 (4 percentage points, $\approx$ 9.8% relative) with parameter and energy reductions of $\approx$ 137.3k ($\approx$ 16.5%) and $\approx$ 19.2% respectively. These consistent parameter reductions across datasets reflect that the pruning operated on fixed structural units (Fire modules) rather than on fine-grained per-dataset re-architecting, which explains the near-constant absolute parameter savings and the dataset-dependent accuracy gains.

Beyond the SqueezeNet family, the results expose a clear capacity–efficiency spectrum. Extremely compact architectures such as *MixNet* and *ShuffleNetV2* attain the lowest measured energies (e.g., MixNet achieves ~0.002–0.009 mJ across datasets) because their parameter counts and multiply–accumulate (MAC) budgets are minimal (MixNet: ~85k parameters on CIFAR-10). However, this economy comes at a measurable cost in representational capacity and thus classification performance: MixNet and ShuffleNetV2 systematically produce lower accuracies, particularly on CIFAR-100 and Tiny ImageNet where inter-class variability and the number of classes magnify the capacity requirement. On the opposite extreme, large models such as *Xception* incur very large MAC counts and consequently dominate the energy column (e.g., Tiny ImageNet Xception's MACs and energy are orders of magnitude higher than other models), yet they do not produce proportionally higher SNN accuracies; this suggests that naive up-scaling of parameter count does not translate to better spiking-domain performance and may introduce optimization difficulties (sparse spike propagation, vanishing spike activity, or poor adaptation of certain operator patterns). Mid-sized, mobile-oriented networks such as *MnasNet* produce a favorable balance: they retain comparatively high accuracy (e.g., 0.78 on CIFAR-10) with moderate energy, implying that architectures originally designed for mobile efficiency translate reasonably well to the spiking regime. Taken together, these observations suggest a practical guideline for practitioners: when target accuracy is the principal objective under constrained compute, structured pruning of a compact but expressive backbone (as in SqueezeNet-P) yields better overall returns than either extreme lightweighting or naive over-parameterization.

Table 4.3 further quantifies the conversion trade-offs between the conventional CNN domain and its SNN counterpart. Across the evaluated architectures, conversion to spiking implementations yields substantial energy reductions: observed energy sav-

Table 4.3: Accuracy and energy comparison between CNN and SNN models on CIFAR-10.

Model	**Acc (%)**		ΔA	**Energy (mJ)**		$\eta_{\mathbf{E}}$
	CNN	SNN		CNN	SNN	
ShuffleNetV2	70	40	30	**0.010**	0.004	2.5
Xception	79	75	4	2.44	0.444	5.5
MnasNet	**83**	78	5	0.113	0.059	1.9
MixNet	82	73	9	0.047	**0.003**	15.7
SqueezeNet	81	75	6	0.252	0.036	7.0
SqueezeNet-P	82	**81**	1	0.168	0.030	5.6

E = energy consumption measured in millijoules (mJ).
$\eta_{\mathbf{E}} = E_{\text{CNN}}/E_{\text{SNN}}$, indicates how many times more energy the CNN model consumes compared to the corresponding SNN model.
$\Delta\text{A} = \text{Accuracy}_{\text{CNN}} - \text{Accuracy}_{\text{SNN}}$.

ings span from approximately 1.9× (MnasNet) up to ∼ 15.7× (MixNet), with intermediate values for Xception (≈ 5.5×), SqueezeNet (7.0×) and the pruned SqueezeNet-P (≈ 5.6×). These energy reductions are achieved with largely modest accuracy penalties: for most networks the SNN loses between 1 and 9 percentage points compared to its CNN baseline. The notable exception is ShuffleNetV2, which suffers a severe degradation (CNN 70% → SNN 40%, a 30 percentage point drop). This outlier highlights an important architectural interaction: operators that rely heavily on fine-grained channel rearrangements or group convolutions (channel shuffle, grouped convolutions) appear vulnerable to the temporal and sparse nature of spike-based computation, because their discriminatory power depends on dense, real-valued channel mixing that is not trivially preserved after discretization into spike events or after surrogate-gradient training. In contrast, the pruned SqueezeNet-P shows an almost lossless conversion (CNN 82% vs SNN 81%, a single percentage point difference), demonstrating that structured pruning can serve not only as a compression mechanism but also as a conversion facilitator: by removing redundant channels and modules it produces a sparser, more robust feature flow whose activation statistics are easier to represent and train in the spiking domain. From an application perspective, these results imply distinct operational regimes: ultra-low-power, coarse recognition tasks may legitimately select MixNet or similarly tiny architectures (accepting the accuracy trade-off) whereas deployments that require near-CNN performance at reduced energy should prioritize architectures that either were designed for efficient inference (MnasNet) or have undergone careful structured pruning and spiking-aware fine-tuning (SqueezeNet-P). Finally, we note a methodological caveat that should guide

thesis-level claims: these results are reported for the fixed training/conversion pipeline and a single random seed; for final publication-quality claims we recommend reporting means and variances across multiple seeds, layerwise spike activity and per-class error analyses, and where possible, profiling energy at finer granularity (per-layer energy and per-operation spike counts) to fully validate the causal links between architecture, spike dynamics and measured energy.

4.4 Qualitative Analysis

Figure 4.2 shows the training and validation curves for loss and accuracy. Both the training and validation accuracy increase steadily, eventually stabilizing around 80%. The validation loss closely follows the training loss, which indicates that the model generalizes well to unseen data without severe overfitting. Interestingly, the training loss is still on a downward trend at the end of 150 epochs. This suggests that training for more epochs may further improve performance, particularly in reducing the gap between training and validation accuracy.

Additionally, the validation accuracy saturates earlier than the training accuracy, which is a common phenomenon caused by the model learning dataset-specific patterns that do not transfer perfectly to validation samples. The relatively smooth convergence of both curves demonstrates the stability of training and confirms that the model architecture is effective for the given task.

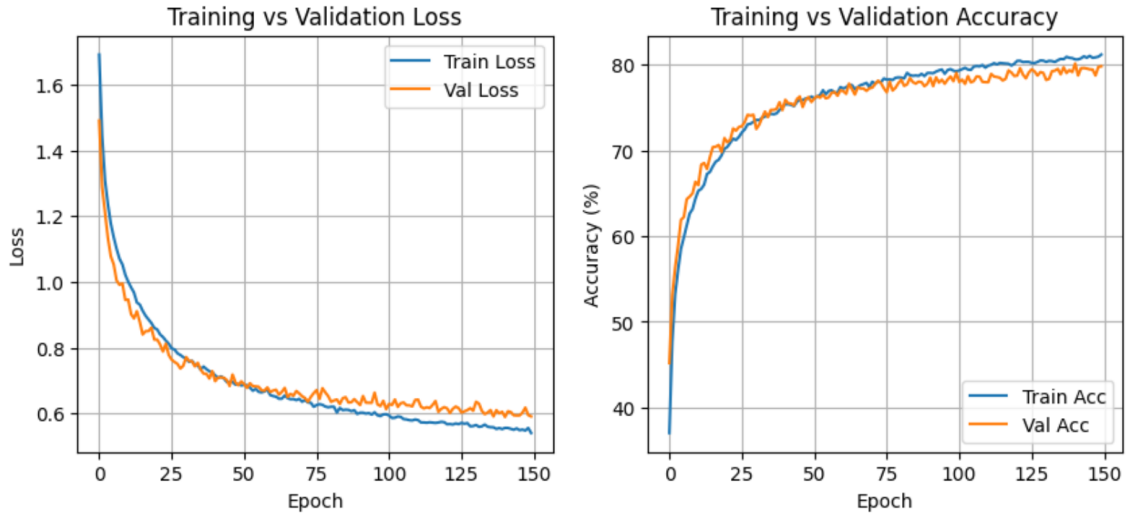


Figure 4.2: Training vs Validation Loss and Accuracy curves for SNN SqueezeNet-P on CIFAR-10. The decreasing loss indicates potential performance improvements with additional epochs.

To better understand the complementary strengths of CNNs and SNNs, we analyze

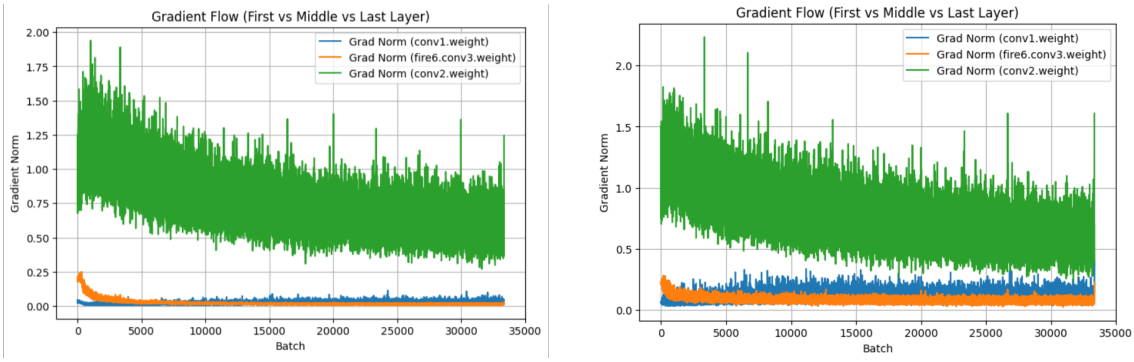
sample predictions from CNN SqueezeNet and the proposed SNN SqueezeNet-P on CIFAR-10 (Table 4.4). Both models correctly recognize well-defined categories such as *bird*, *dog*, and *frog*, but they diverge on visually ambiguous cases. For example, the SNN misclassifies *truck* as *automobile*, likely because spiking neurons emphasize local edge and contour patterns, which are structurally similar between the two classes. In contrast, the CNN misclassifies *horse* as *dog*, a mistake that can be attributed to texture- and color-based similarities, as CNNs rely heavily on spatial appearance features without strong temporal dynamics. These differences highlight how SNNs, with their event-driven processing, tend to confuse objects with overlapping structural shapes, while CNNs are more prone to errors driven by background context or surface-level visual textures. Such complementary error modes suggest that hybrid or ensemble approaches could leverage the robustness of both representations.

Table 4.4: Qualitative comparison of CNN SqueezeNet and SNN SqueezeNet-P on CIFAR-10. Correct predictions are shown normally, while incorrect predictions are highlighted in red.

Image 1	**Image 2**	**Image 3**	**Image 4**	**Image 5**
				
Ground Truth: Bird	Ground Truth: Dog	Ground Truth: Frog	Ground Truth: Truck	Ground Truth: Horse
CNN: Bird	CNN: Dog	CNN: Frog	CNN: Truck	CNN: Dog
SNN: Bird	SNN: Dog	SNN: Frog	SNN: Automobile	SNN: Horse

Figure 4.3 provides a comparative visualization of gradient propagation in the SNN SqueezeNet architecture, with and without pruning. In the baseline model (Figure 4.3(a)), the gradient norms in both the initial and middle layers rapidly diminish to values approaching zero. This phenomenon is indicative of severe vanishing gradient effects, whereby early and intermediate feature extraction modules receive negligible learning signals during backpropagation. As a result, weight updates in these layers are effectively stalled, leading to suboptimal representation learning and sluggish convergence.

By contrast, the pruned variant (Figure 4.3(b)) exhibits a markedly improved gradient distribution. Notably, the gradient magnitudes in the initial and middle layers remain consistently above zero, ensuring that these modules actively participate in the optimization process. This preservation of non-trivial gradients implies that pruning acts not merely as a structural compression technique but also as a facilitator of more stable gradient flow. The removal of redundant or low-contributing connections reduces inhibitory accumulation, thereby enhancing signal propagation across layers.



(a) SNN SqueezeNet **(b)** SNN SqueezeNet-P

Figure 4.3: Comparison of gradient flow across layers in (a) the base SNN SqueezeNet and (b) the SNN SqueezeNet-P. Pruning alleviates the vanishing gradient problem by redistributing gradient flow, allowing earlier and middle layers to receive stronger and more stable updates during training.

These findings suggest that pruning serves a dual role: reducing model complexity while simultaneously alleviating the vanishing gradient problem. Consequently, the pruned SNN SqueezeNet demonstrates more effective weight adaptation, faster convergence, and improved generalization compared to its unpruned counterpart. The results underscore the importance of architectural sparsification as a strategy for stabilizing gradient dynamics in deep spiking neural networks.

4.5 Challenges and Limitations

While the results presented in this chapter are promising, several challenges and limitations should be acknowledged. First, the performance of the proposed models on large-scale datasets such as Tiny ImageNet remains modest, with accuracies below 50%. This highlights the difficulty of scaling spiking neural networks to more complex visual recognition tasks, where the limited representational power of lightweight architectures and the sparsity constraints of spiking activity can become bottlenecks. Furthermore, in practice, device-level imperfections, synaptic noise, and event-driven communication overheads may alter the real-world efficiency gains, potentially reducing the magnitude of energy savings observed in simulation.

Another important limitation arises from computational resource constraints. Due to restricted access to high-performance hardware, extensive hyperparameter tuning and longer training schedules could not be explored. As a result, the models may not have reached their full performance ceiling, and comparison with state-of-the-art baselines should be interpreted with caution. Additionally, only a limited set of optimization techniques, such as pruning, were investigated, whereas complementary

strategies including knowledge distillation, quantization-aware training, or hybrid spike-rate coding could further improve performance. The project was also bounded by time and resource constraints, preventing large-scale ablation across different spiking neuron models, coding schemes, or more diverse datasets beyond CIFAR and Tiny ImageNet. Collectively, these limitations underline that while the findings provide strong empirical evidence for pruning as an optimization strategy, there remains ample room for deeper exploration and refinement.

4.6 Implications and Significance

Despite the above constraints, the results of this study carry significant implications for the field of neuromorphic computing. The experimental evidence shows that structured pruning can not only compress spiking models but also improve their generalization by removing redundant connections, thus making SNNs both leaner and more robust. Importantly, the comparative analysis with conventional CNNs demonstrates that SNNs, when optimized through pruning, can approach or even match CNN-level accuracy while delivering substantial reductions in energy consumption. This strengthens the case for SNNs as viable alternatives to traditional deep learning models in energy-constrained scenarios.

The broader significance of these findings lies in their applicability to real-world deployment contexts such as edge AI, Internet of Things (IoT) devices, and neuromorphic hardware platforms. In these domains, energy efficiency is not a secondary consideration but a primary design constraint, making pruned SNNs particularly attractive. Moreover, the study establishes pruning as a practical optimization strategy that can be integrated into future neuromorphic pipelines alongside other techniques such as quantization and event-driven inference. By demonstrating that pruning improves both efficiency and accuracy under spiking constraints, this work highlights a pathway toward scalable, deployable neuromorphic systems that bridge the gap between academic research and practical applications.

Chapter 5

Conclusion

This concluding chapter encapsulates the core findings, contributions, and broader implications of the research on lightweight CNN-to-SNN conversion for energy-efficient image classification. By revisiting the research objectives, summarizing key results, discussing their significance, acknowledging limitations, and proposing future directions, this chapter provides a reflective synthesis of the work and its place within the field of neuromorphic computing.

5.1 Restating Research Objectives and Questions

The primary objective of this thesis was to develop and evaluate a systematic methodology for converting lightweight Convolutional Neural Networks (CNNs) into Spiking Neural Networks (SNNs) to achieve energy-efficient image classification suitable for resource-constrained edge devices. The research was guided by three central questions: (1) How effectively can lightweight CNN architectures be converted to SNNs while preserving classification accuracy? (2) Can structured pruning enhance the performance and efficiency of these converted SNNs? (3) What are the trade-offs between accuracy, computational complexity, and energy consumption in CNN-to-SNN conversions across datasets of varying complexity? These questions framed the study, directing the design of the methodology, the selection of benchmark datasets (CIFAR-10, CIFAR-100, and Tiny ImageNet), and the evaluation framework.

5.2 Summary of Key Findings

The study successfully demonstrated that lightweight CNNs, specifically ShuffleNet, MnasNet, MixNet, and SqueezeNet, can be converted into SNNs with significant energy savings, albeit with varying degrees of accuracy degradation. Among these, the SNN variant of SqueezeNet emerged as the most promising base architecture, achieving a favorable balance between accuracy, parameter count, and spike efficiency. Through random searching hyperparameters and structured pruning, the optimized SNN-SqueezeNet-P model further improved performance, achieving up to 81% accuracy on CIFAR-10 (a 6% improvement over the unpruned SNN-SqueezeNet), while reducing parameters by approximately 18.7% and energy consumption by 16.9%. Similar trends were observed on CIFAR-100 and Tiny ImageNet, with accuracy gains of 7% and 4%, respectively, alongside consistent reductions in parameters and energy. Compared to their CNN counterparts, SNNs exhibited substantial energy efficiency improvements, with SNN-SqueezeNet-P reducing energy consumption by 5.6× on CIFAR-10 while maintaining near-CNN accuracy (81% vs. 82%). The ablation studies underscored that pruning early-stage Fire modules preserved critical discriminative features, enhancing both accuracy and efficiency. These findings confirm that well-designed conversion and pruning strategies can produce SNNs that approach CNN-level performance while drastically lowering energy costs.

5.3 Limitations and Key Findings

The results contribute significantly to the field of neuromorphic computing by demonstrating that lightweight CNN-to-SNN conversion, coupled with structured pruning, can bridge the accuracy-efficiency gap that has historically limited SNN adoption. Theoretically, this work advances the understanding of how spiking dynamics interact with lightweight architectures, showing that sparsity in spiking computation can be leveraged to enhance both efficiency and generalization when redundant pathways are carefully removed. Methodologically, the proposed pipeline encompassing dataset selection, ANN-to-SNN conversion, architecture evaluation, pruning, and comprehensive benchmarking offers a reproducible framework for future SNN research. Empirically, the study provides robust evidence that SNNs can achieve practical tradeoffs, making them viable for edge intelligence applications where energy constraints are paramount.

These findings resonate with prior work in the field, particularly studies like Rueckauer et al. [29] and Han et al. [8], which emphasized conversion techniques to pre-

serve accuracy. However, by focusing on lightweight architectures and integrating pruning, this research extends the applicability of SNNs to resource-constrained environments, addressing gaps identified in the literature review. The results also align with the broader push toward sustainable AI, where energy-efficient models are increasingly critical for large-scale IoT deployments and environmentally conscious computing.

This thesis makes several distinct contributions to the field of neuromorphic computing:

1. **Systematic Evaluation Framework**: The study establishes a comprehensive methodology for comparing lightweight CNNs and their SNN counterparts across multiple dimensions (accuracy, F1-score, parameters, ACs/MACs, and energy consumption), providing a robust benchmark for future research.
2. **Pruning Strategy for SNNs**: The tailored pruning approach for SNN-SqueezeNet demonstrates that structured removal of Fire modules can simultaneously improve accuracy (by up to 6% on CIFAR-10) and reduce computational overhead (by $\sim$18.7% in parameters and $\sim$16.9% in energy), offering a novel optimization technique for spiking architectures.
3. **Energy Efficiency Insights**: The results quantify significant energy savings (up to 15.7$\times$ for MixNet, 5.6$\times$ for SqueezeNet-P) with minimal accuracy loss, reinforcing SNNs as practical alternatives for edge deployment.
4. **Practical Guidelines**: By identifying SNN-SqueezeNet-P as a Pareto-optimal solution, the study provides actionable design principles for deploying energy-efficient neural networks on neuromorphic hardware, emphasizing the importance of architecture selection and pruning.

These contributions advance both the theoretical understanding of SNNs and their practical deployment, particularly in energy-sensitive applications such as IoT, wearables, and autonomous systems.

Despite its contributions, the study faces several limitations. First, the performance of SNNs on complex datasets like Tiny ImageNet remains modest (45% accuracy for SNN-SqueezeNet-P), indicating challenges in scaling lightweight SNNs to large-scale tasks. Second, energy estimates were derived from idealized models based on Horowitz (2014) [11], which may not fully capture real-world neuromorphic hardware constraints such as synaptic noise or communication overheads. Third, computational resource constraints limited the scope of hyperparameter tuning and training duration,

potentially preventing models from reaching their full potential. Additionally, only a subset of optimization techniques (e.g., pruning) was explored, leaving complementary methods like quantization or knowledge distillation unaddressed. Finally, the study focused on a fixed training pipeline with a single random seed, which may not fully account for variability in performance. These limitations suggest that while the findings are robust within the study's scope, further refinement is needed for broader applicability.

The limitations and findings of this study point to several promising directions for future research:

1. **Scaling to Complex Datasets**: Investigating advanced training techniques, such as hybrid spike-rate coding or knowledge distillation, could improve SNN performance on large-scale datasets like ImageNet or real-world sensor streams.
2. **Hardware-in-the-Loop Evaluation**: Validating energy estimates on actual neuromorphic hardware (e.g., Intel Loihi, Innatera Pulsar) would bridge the simulation-to-hardware gap, accounting for practical constraints like memory bandwidth and spike routing.
3. **Expanded Optimization Strategies**: Exploring complementary optimization techniques, such as quantization-aware training or dynamic threshold adaptation, could further enhance SNN efficiency and accuracy.
4. **Robustness Analysis**: Conducting multi-seed experiments, per-class error analyses, and layerwise spike activity profiling would provide deeper insights into the robustness and behavior of SNNs.
5. **Domain-Specific Applications**: Extending the methodology to domain-specific tasks (e.g., biomedical signal processing, audio detection) could validate SNNs in diverse, real-world edge scenarios.

These directions offer opportunities to refine and extend the proposed framework, addressing both the theoretical and practical challenges of neuromorphic computing.

This research journey has been both challenging and rewarding, revealing the complexities of balancing biological plausibility, computational efficiency, and predictive performance in SNNs. The unexpected finding that pruning not only reduces energy consumption but also enhances accuracy underscores the potential of optimization strategies to reshape spiking architectures. The process of navigating conversion challenges, dataset variability, and computational constraints has deepened our understanding of neuromorphic systems and their practical viability. This study repre-

sents a step toward sustainable AI, demonstrating that SNNs can deliver robust performance in energy-constrained environments, with broader implications for the future of edge intelligence.

This thesis demonstrates that lightweight CNN-to-SNN conversion, enhanced by structured pruning, offers a viable path to energy-efficient image classification without significant sacrifices in accuracy. By achieving up to 15.7× energy savings and near-CNN performance with SNN SqueezeNet-P, the research establishes a compelling case for SNNs in edge computing applications. The proposed methodology and empirical benchmarks provide a foundation for future advancements in neuromorphic computing, paving the way for scalable, sustainable, and intelligent systems that can operate seamlessly in resource-constrained environments. As the field progresses, this work serves as both a milestone and a catalyst for further innovation in energy-efficient deep learning.

References

[1] J. Acharya and A. Basu, "Neuromorphic spiking neural network algorithms," in *Handbook of Neuroengineering*, N. V. Thakor, Ed. Singapore: Springer Nature Singapore, 2020, pp. 1–37, ISBN: 978-981-15-2848-4. DOI: 10.1007/978-981-15-2848-4_44-1 [Online]. Available: https://doi.org/10.1007/978-981-15-2848-4_44-1

[2] M. Z. Alom et al., *The history began from alexnet: A comprehensive survey on deep learning approaches*, 2018. arXiv: 1803.01164 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1803.01164

[3] L. Deng and Q. Gu, "Optimal brain surgeon for spiking neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[4] L. Deng, Q. Gu, Y. Zhao, M. Hu, and Y. Wang, "Temporal efficient training for spiking neural networks," in *International Conference on Learning Representations*, 2022.

[5] 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.

[6] W. Fang, Y. Wang, Z. Li, C. Liu, Z. Fang, and L. Ming, "Incorporating learnable surrogate gradient for direct training of spiking neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 7920–7928.

[7] T. Guide, "We're building chips that think like the brain––neuromorphic computing in smart devices," *Tom's Guide*, 2025, Innatera's Pulsar neuromorphic chip for edge devices.

[8] B. Han, G. Srinivasan, and K. Roy, "Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13 555–13 564. DOI: 10.1109/CVPR42600.2020.01357

[9] S. Han, J. Pool, J. Tran, and W. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," in *International Conference on Learning Representations (ICLR)*, 2016.

[10] K. He, X. Zhang, S. Ren, and J. Sun, *Deep residual learning for image recognition*, 2015. arXiv: 1512.03385 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1512.03385

[11] 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 (ISSCC)*, IEEE, 2014, pp. 10–14. DOI: 10.1109/ISSCC.2014.6757323 [Online]. Available: https://ieeexplore.ieee.org/document/6757323

[12] A. G. Howard et al., "Mobilenets: Efficient convolutional neural networks for mobile vision applications," in *arXiv preprint arXiv:1704.04861*, 2017.

[13] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size*, 2016. arXiv: 1602.07360 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1602.07360

[14] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, *Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5MB model size*, 2017. [Online]. Available: https://openreview.net/forum?id=S1xh5sYgx

[15] S. R. Kheradpisheh, E. Ghourchian, T. Masquelier, S. J. Thorpe, and T. Masquelier, "Stdp-based spiking deep neural networks for object recognition," in *arXiv preprint arXiv:1805.11675*, 2018.

[16] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Neural Information Processing Systems*, vol. 25, Jan. 2012. DOI: 10.1145/3065386

[17] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, pp. 598–605, 1990.

[18] E. Ledinauskas, J. Ruseckas, A. Juršėnas, and G. Buračas, *Training deep spiking neural networks*, 2020. arXiv: 2006.04436 [cs.NE]. [Online]. Available: https://arxiv.org/abs/2006.04436

[19] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *International Conference on Learning Representations (ICLR)*, 2017.

[20] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan, and C. Zhang, "Learning efficient convolutional networks through network slimming," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2736–2744.

[21] P. Merolla et al., *A million spiking-neuron integrated circuit with a scalable communication network and interface*, en, 2014. [Online]. Available: https://www.semanticscholar.org/paper/A-million-spiking-neuron-integrated-circuit-with-a-Merolla-Arthur/680a38e8f025685b192e9e0cf755c6b664963551

[22] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 11 264–11 272.

[23] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *International Conference on Learning Representations (ICLR)*, 2017.

[24] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate Gradient Learning in Spiking Neural Networks," *arXiv preprint arXiv:1901.09948*, 2019. [Online]. Available: https://arxiv.org/abs/1901.09948

[25] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019. DOI: 10.1109/MSP.2019.2931595

[26] H. C. V. Ngu and K. M. Lee, "Effective conversion of a convolutional neural network into a spiking neural network for image recognition tasks," *Applied Sciences*, vol. 12, no. 11, 2022, ISSN: 2076-3417. DOI: 10.3390/app12115749 [Online]. Available: https://www.mdpi.com/2076-3417/12/11/5749

[27] M. K. Pasupuleti, "Spiking neural networks for energy-efficient edge intelligence," *IJAIRI*, 2025.

[28] K. Roy, A. Jaiswal, and P. Panda, "Towards spike-based machine intelligence with neuromorphic computing," in *Nature*, vol. 575, Nature Publishing Group, 2019, pp. 607–617.

[29] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017. DOI: 10.3389/fnins.2017.00682 [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnins.2017.00682/full

[30] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, "Going deeper in spiking neural networks: Vgg and residual architectures," *Frontiers in Neuroscience*, vol. 13, p. 153, 2019. DOI: 10.3389/fnins.2019.00153 [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnins.2019.00153/full

[31] J. Shen, Q. Xu, J. K. Liu, Y. Wang, G. Pan, and H. Tang, *Esl-snns: An evolutionary structure learning strategy for spiking neural networks*, 2023. arXiv: 2306.03693 [cs.NE]. [Online]. Available: https://arxiv.org/abs/2306.03693

[32] J. Shen, Q. Xu, G. Pan, and B. Chen, *Improving the sparse structure learning of spiking neural networks from the view of compression efficiency*, 2025. arXiv: 2502.13572 [cs.HC]. [Online]. Available: https://arxiv.org/abs/2502.13572

[33] X. Shi, J. Ding, Z. Hao, and Z. Yu, "Towards energy efficient spiking neural networks: An unstructured pruning framework," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=eoSeaK4QJo

[34] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2015. arXiv: 1409.1556 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1409.1556

[35] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, 2015. arXiv: 1409.1556 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1409.1556

[36] M. Tan and Q. V. Le, "Mixnet: Mixed depthwise convolutional kernels," *arXiv preprint arXiv:1907.09595*, 2019.

[37] M. Tan et al., "Mnasnet: Platform-aware neural architecture search for mobile," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2820–2828.

[38] Unnamed, "Spiking neural networks and their applications: A review," *Brain Sciences*, 2022.

[39] H. Wang, B. Kim, J. Xie, and Z. Han, "Energy drain of the object detection processing pipeline for mobile devices: Analysis and implications," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 1, pp. 41–60, 2021. DOI: 10.1109/TGCN.2020.3041666

[40] D. Wu et al., "Build an energy-efficient, accurate spiking neural network," *Frontiers in Neuroscience*, 2022.

[41] D. Xu et al., *Edge intelligence: Architectures, challenges, and applications*, 2020. arXiv: 2003.12172 [cs.NI]. [Online]. Available: https://arxiv.org/abs/2003.12172

[42] T.-J. Yang, Y.-H. Chen, and V. Sze, *Designing energy-efficient convolutional neural networks using energy-aware pruning*, 2017. arXiv: 1611.05128 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1611.05128

[43] T.-J. Yang, A. Howard, A. Smola, B. Singh, S. Chien, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5687–5695.

[44] Y.-X. Zeng, J.-W. Hsieh, X. Li, and M.-C. Chang, *Mixnet: Toward accurate detection of challenging scene text in the wild*, 2023. arXiv: 2308.12817 [cs.CV]. [Online]. Available: https://arxiv.org/abs/2308.12817

[45] X. Zhang, X. Zhou, M. Lin, and J. Sun, *Shufflenet: An extremely efficient convolutional neural network for mobile devices*, 2017. arXiv: 1707.01083 [cs.CV]. [Online]. Available: https://arxiv.org/abs/1707.01083

[46] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of CVPR*, 2018, pp. 6848–6856.

[47] A. Zheng et al., "Going deeper with directly-trained larger spiking neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 6677–6685.