A Case Study in Practical Neuromorphic Computing: Heartbeat Classification on the Loihi Neuromorphic Processor

by

Kyle Buettner

B.S. Computer Engineering, University of Pittsburgh, 2019

Submitted to the Graduate Faculty of

the Swanson School of Engineering in partial fulfillment

of the requirements for the degree of

Master of Science in Electrical and Computer Engineering

University of Pittsburgh

2021

UNIVERSITY OF PITTSBURGH

SWANSON SCHOOL OF ENGINEERING

This thesis was presented

by

Kyle Buettner

It was defended on

March 29, 2021

and approved by

Murat Akcakaya, Ph.D., Associate Professor, Department of Electrical and Computer Engineering

Rajkumar Kubendran, Ph.D., Assistant Professor, Department of Electrical and Computer Engineering

Thesis Advisor: Alan D. George, Ph.D., Professor, Department Chair, R&H Mickle Endowed Chair, Department of Electrical and Computer Engineering

A Case Study in Practical Neuromorphic Computing: Heartbeat Classification on the Loihi Neuromorphic Processor

Kyle Buettner, M.S.

University of Pittsburgh, 2021

One potential method of efficiently deploying deep neural networks is with neuromorphic computing, a paradigm of processing that emulates the energy-efficient spiking neural networks (SNNs) of the human brain. This research evaluates artificial-to-spiking neural network (ANN-to-SNN) conversion as a practical methodology to perform energy-efficient heartbeat classification on the state-of-the-art Intel Loihi neuromorphic processor. In particular, a spiking 1D-convolutional neural network (1D-CNN) model is designed through ANN-to-SNN conversion with *SNN-Toolbox* to identify arrhythmias on Loihi. Insights into the conversion process are gained through experimentation with accuracy-latency tradeoffs, neuron reset mechanisms, and weight and bias values. These insights enable the spiking 1D-CNN to be optimized for low latency and high accuracy on Loihi, and then compared to an architecturally identical artificial neural network (ANN) on Intel Core i7 CPU, Intel Neural Compute Stick 2, and Google Coral Edge TPU devices in terms of accuracy, latency, and energy performance. Across five classes, the spiking 1D-CNN is found to reach an accuracy and macro-averaged F1 score of 97.8% and 87.9%, respectively, compared to 98.4% and 90.8% for the ANN. Additionally, with the lowest dynamic power across devices, Loihi is estimated to provide a 28 times lower energy-delay product for the model versus the CPU baseline. However, with the highest latency across devices, Loihi is also estimated to result in a 1.5 times and 110 times higher energy-delay product versus the Intel Neural Compute Stick 2 and Google Coral Edge TPU, respectively. This higher latency is determined to result from x86 core-to-host I/O and x86 core-based management bottlenecks. From these findings, insights are provided regarding future directions for practical neuromorphic computing.

Table of Contents

List of Tables

List of Figures

Preface

This work was supported by SHREC industry and agency members and by the IUCRC Program of the National Science Foundation under Grant No. CNS-1738783.

The author would like to thank Dr. Alan D. George for his guidance as project advisor.

The author would like to thank Dr. Ryad Benosman and Dr. Rajkumar Kubendran for providing insights into neuromorphic engineering.

Thanks is given to the members of Intel Labs, who provided access to Loihi hardware and gave valuable feedback during the course of this research.

Much thanks is also deserved to the graduate student researchers in the SHREC lab for helpful feedback and ideas.

1.0 Introduction

A major consideration in the deployment of deep neural networks (DNNs) at the edge is energy efficiency, as DNNs may be constrained to perform inference with milliwatt or microwatt power budgets [1]. The energy efficiency of neural network inference significantly depends on the choice of processor. Therefore, the evaluation of low-power neural network architectures is a crucial area of research to provide insights for applications at the edge.

Neuromorphic computing is an attractive paradigm for energy-efficient processor design as it uses the human brain's high parallelism and low-power form as inspiration. The emergence of Intel's neuromorphic research chip, Loihi [2], a platform for spiking neural networks (SNNs), provides the opportunity to evaluate a neuromorphic platform versus more standard processing architectures. Additionally, it enables the exploration of applications that may benefit from the energy efficiency of neuromorphic computing. This research identifies heartbeat classification through electrocardiogram signals as a fitting case study, as it is hypothesized that an SNN can perform low-latency, low-energy identification of arrhythmias on Loihi. An energy-efficient design could be deployed on a wearable device to provide real-time insights and help in diagnosis and prevention of heart disease, a top health challenge in the USA [3].

The design of SNNs, especially with regards to achieving similar accuracy to conventional artificial neural network (ANNs), is a growing area of research. Therefore, to perform heartbeat classification on Loihi, this research investigates artificial-to-spiking neural network (ANN-to-SNN) conversion with the *SNN-Toolbox* framework [4] as an SNN design methodology. A variety of considerations are explored in the practical design of a spiking 1D-convolutional neural network (1D-CNN). Overall, this methodology investigation elucidates strategies for accurate and high-performance SNN design.

Furthermore, the latency and energy performance of an optimized spiking 1D-CNN on Loihi is compared to the performance of an architecturally identical ANN on Intel Core i7 CPU, Intel Neural Compute Stick 2, and Google Coral Edge TPU devices. Through benchmarking, this research gauges how a neuromorphic approach for energy-efficient heartbeat

classification compares to approaches on other architectures for neural network inference. This study also offers insights regarding future directions for the exploration of neuromorphic computing in low-energy use cases.

2.0 Background

Background for this research is presented as three sections. First, electrocardiogram analysis is discussed to give motivation for heartbeat classification. Second, the foundations of neuromorphic computing and SNNs are described. Lastly, this section concludes with an overview of the Loihi architecture.

2.1 Electrocardiogram Analysis

Electrocardiograms (ECGs or EKGs) are tests that capture electrical activity fluctuations in the heart [5]. Electrical activity changes when heart muscles polarize in a rhythmic manner, creating heartbeats. Heartbeats are represented in ECG signals by morphological features such as the QRS complex, P wave, and T wave. To record ECG signals, noninvasive electrodes (or leads) are attached to the skin, often in 5-lead or 12-lead configurations. In a medical setting, signals can be analyzed by clinicians to characterize heart function and diagnose conditions related to cardiac health, such as arrhythmias. It is also possible to perform portable, continuous recording or monitoring of ECG signals for several hours or days using a wearable device, such as a Holter monitor [6]. With portable devices, signals can be saved for later analysis, or anomalous heartbeats can be detected automatically in real time, providing immediate insights that can be used to trigger additional recording or alerts. These insights can ultimately result in improved opportunities for care.

2.2 Neuromorphic Computing

Neuromorphic computing is a type of processing that closely models the human brain. Carver Mead helped formalize neuromorphic processors in the 1980s with a book on Very Large-Scale Integration (VLSI) of analog neural circuits [7]. The field has since developed to encapsulate digital and analog modeling of SNNs.

A neuron in an SNN has a membrane potential that is augmented over time through the presentation of binary spikes on synapses. Once a voltage threshold is reached, that neuron fires a spike and reset its state. Different neuron models can represent spiking behavior, with one example being the leaky integrate-and-fire (LIF) model, which treats neuron membranes as leaky integrator RC circuits [8]. Moreover, information can be captured in the rate or timing of spikes. These codings are called "rate coding" and "temporal coding", respectively.

SNN energy is primarily consumed through spiking, which is asynchronous and sparse. This property has inspired designers to strive to create energy-efficient neuromorphic hardware. Such a processor provides a mode of operation in contrast to the energy-consuming synchronous clocking of CPUs and GPUs. Neuromorphic hardware also ideally avoids the processor-memory bottleneck of von Neumann architectures, where application throughput is limited by processor-memory bandwidth. This benefit is a result of spiking neurons maintaining local state, thus co-locating computation and memory.

2.3 The Loihi Neuromorphic Chip

A recent neuromorphic research chip is Loihi, a digital integrated circuit fabricated on a 14nm process [2]. Loihi represents a key step forward in neuromorphic design, especially due to its programmable on-chip learning engine, a unique capability versus past designs such as IBM TrueNorth [9]. Each Loihi chip contains a mesh of 128 neuron cores with 1,024 neurons per core and up to 130 million synapses for spike communication. Additionally, each Loihi chip contains three embedded x86 Lakemont cores, where sequential neural interacting processes (SNIPs) are run to manage neuron cores. Loihi chips come in a variety of form factors, ranging from the USB device Kapoho Bay (1-2 chips) to the recently introduced high-performance computing system Pohoiki Springs (768 chips) [10].

Loihi neuron dynamics are based on current-based synapse (CUBA) LIF modeling with discretely approximated variables for membrane potential and synaptic response current. In reference to equations from [2], each Loihi neuron receives an input spike train over time t. Each individual spike is referred to as spike k. The equation for a spike train is the sum of Dirac delta functions, shown in Equation 2-1.

$$\sigma(t) = \sum_{k} \delta(t - t_k) \tag{2-1}$$

These spike trains are filtered using the synaptic filter impulse response $\alpha_u(t)$, shown in Equation 2-2 with time constant τ_u and Heaviside function $H(t)$.

$$\alpha_u(t) = \tau_u^{-1} \exp\left(\frac{-t}{\tau_u}\right) H(t) \tag{2-2}$$

The filtered responses are weighted by w_{ij}, the weight of the synapse between neurons i and j, and added with a bias term b_i to produce synaptic response current for a neuron i, shown in Equation 2-3.

$$u_i(t) = \sum_{j \neq i} w_{ij} (\alpha_u * \sigma_j)(t) + b_i \tag{2-3}$$

The integration of $u_i(t)$ produces the membrane potential for neuron i, as shown in Equation 2-4, with V_{t_i} representing the voltage threshold and τ_v representing a time constant.

$$\dot{v_i}(t) = \frac{-v_i(t)}{\tau_v} - V_{t_i}\sigma_i(t) + u_i(t) \tag{2-4}$$

3.0 Related Research

This research details three areas of related work. First, a survey of ECG-based heartbeat classification is provided, with emphasis on SNN approaches. Second, methods and frameworks to design SNNs, including *SNN-Toolbox*, are described. Lastly, this section concludes with past performance comparisons of SNNs on Loihi versus ANNs on other devices.

3.1 Electrocardiogram-Based Heartbeat Classification

Multiple stages can exist in an ECG analysis pipeline: signal preprocessing, heartbeat segmentation, feature extraction, and classification [11]. One of the most common datasets used to train analysis pipelines is the MIT-BIH Arrhythmia Database, which contains 30 minute ECG recordings for 47 patients with various arrhythmia types [12]. The Association for the Advancement of Medical Instrumentation (AAMI) provides standards for model evaluation on such datasets [13].

A variety of machine learning models have been used for heartbeat classification. Examples include 1D-CNNs [14] and long short-term memory models (LSTMs) [15]. SNNs have also been recently explored for ECG analysis, as a major consideration with portable monitoring is energy efficiency [16]. For example, Corradi et al. implement a spiking recurrent neural network on the custom VLSI-based Dynamic Neuromorphic Asynchronous Processor (DYNAP) system, attaining 95% classification accuracy over 18 classes in the MIT-BIH dataset [17]. In another approach on DYNAP, Bauer et al. use reservoir computing to perform ECG-based anomaly detection, with the approach estimated to be sub-mW power [18]. While these works on DYNAP demonstrate promise, there is additional opportunity to evaluate novel SNN designs (such as those through ANN-to-SNN conversion) on more general-purpose and novel neuromorphic hardware (such as Loihi). Lastly, Amirshahi and Hashemi use a reward-modulated spike-timing dependent plasticity (STDP) learning rule to train a spiking ECG classifier [19]. They provide benchmarks of other neural network heart-

beat classifiers on an ARM Cortex-A53 CPU and estimate SNN energy to range from two to nine orders of magnitude smaller. While this work provides some comparative benchmarks, SNN energy is only estimated through simulation and thus does not consider real-world constraints of neuromorphic hardware, such as I/O.

3.2 Spiking Neural Network Design

Deep SNNs have been challenging to accurately train as they are unable to directly use backpropagation [20]. This inability stems from the non-differentiable nature of a spiking neuron's transfer function. Recent work has aimed to overcome this challenge. One example is the SLAYER algorithm [21], which uses temporal credit assignment to distribute error for SNN training. Another example is the Nengo framework [22], which trains an ANN with a differentiable, rate-based approximation of a spiking neuron model and then converts that ANN to an equivalent SNN.

The focus of this research is on an alternative ANN-to-SNN conversion framework, *SNN-Toolbox* [4]. This framework operates by converting each artificial neuron to a spiking integrate-and-fire neuron and computing SNN parameters that best correlate ANN activation values to average SNN firing rates. Regarding the theory behind this framework (with equations introduced in [4]), an ANN ReLU activation for a neuron i in layer l can be modeled as Equation 3-1.

$$a_i^l = \max(0, \sum_{j=1}^{M^{l-1}} W_{ij}^l a_j^{l-1} + b_i^l) \tag{3-1}$$

The values W_{ij}^l and b_i^l respectively refer to the weights between neurons and biases in each layer. The number of neurons in a layer is M^l.

The spiking neurons follow an alternative model, where there is an input current $z_i^l(t)$ for each neuron i, as shown in Equation 3-2.

$$z_i^l(t) = V_{thr}(\sum_{j=1}^{M^{l-1}} W_{ij}^l \Theta_{t,j}^{l-1} + b_i^l) \tag{3-2}$$

V_{thr} represents the voltage threshold for the spiking neurons. Included is also a step function Θ, in which the value of Θ is zero if the input is less than zero and one if the input is greater than or equal to zero. This quantity is further represented in Equation 3-3 as a function of the membrane potential $V_i^l(t)$, input current $z_i^l(t)$, and voltage threshold V_{thr}.

$$\Theta_{t,i}^l = \Theta(V_i^l(t-1) + z_i^l(t) - V_{thr}) \tag{3-3}$$

SNN-Toolbox attempts to correlate the firing rate $r_i^l(t)$ (Equation 3-4) of each spiking neuron i to the activation a_i^l of each artificial neuron.

$$r_i^l(t) = \frac{\sum_{t'=1}^{t} \Theta_{t',i}^l}{t} \tag{3-4}$$

For more information on the theoretical foundations behind how *SNN-Toolbox* relates rates and activations, please refer to [4].

A key strategy employed in this methodology to achieve high accuracy is data-based weight normalization, which avoids overly high firing rates by scaling weights and biases with the maximum activation values calculated over a subset of data. Moreover, analog data can be directly fed into the first hidden layer of an SNN as constant bias current, avoiding the need to encode input data as spikes. In general, *SNN-Toolbox* allows users to transfer models from deep learning frameworks like TensorFlow (TF) and supports TF layer types such as softmax, batch normalization, and pooling. Models converted in this framework can be adapted to fit the constraints of various neuromorphic backends, including Loihi.

3.3 Spiking Neural Network Performance on Loihi

Prior Loihi benchmarking has focused on dynamic energy per inference, the product of dynamic power and latency. For instance, Blouw et al. create a two hidden-layer perceptron for keyword spotting with Nengo on Loihi's Wolf Mountain board [23]. Loihi was estimated to provide 5.3× and 20.5× better dynamic energy per unbatched inference versus the Intel Neural Compute Stick 1 (Movidius) and NVIDIA Jetson TX1 GPU, respectively. Other work

has evaluated Loihi with energy-delay product (EDP), the product of energy and latency, which accounts for slow, low-energy models. For example, Ceolini et al. use SLAYER to create an SNN with three convolutional layers, two pooling layers, and two fully connected layers for gesture recognition [24]. The SNN on Loihi achieved accuracy superior to non-spiking models with fused dynamic vision sensor and electromyography data. Loihi also provided an EDP estimated to be 26× better than that of an NVIDIA Jetson Nano GPU. While Loihi has shown improvements in dynamic energy metrics versus some edge devices, there is additional opportunity to benchmark Loihi versus more novel architectures for neural network inference at the edge, such as the Intel Neural Compute Stick 2 and Google Coral Edge TPU.

4.0 Methodology

The development of neuromorphic platforms like Loihi provides opportunities for competitive benchmarking and exploration of energy-efficient SNN designs. This research explores ANN-to-SNN conversion through *SNN-Toolbox* to perform heartbeat classification on Loihi. While research exists outlining design considerations of *SNN-Toolbox* for MNIST, CIFAR-10, and DvsGesture datasets [25], the performance of *SNN-Toolbox* models on Loihi versus models on edge neural network hardware has yet to be comprehensively explored. The methodology of this research aims to bridge *SNN-Toolbox* design considerations to practical deployment and to compare approaches across neural network hardware.

4.1 Use Case

The use case for this study is ECG-based heartbeat classification on a wearable device. Wearable devices often contain resource-constrained hardware due to a low power budget, making them ideal candidates for low-power neuromorphic computing. A model that runs on a neuromorphic device can potentially be more energy-efficient than a model on a more traditional architecture. This research hypothesizes that Loihi in particular can enhance dynamic energy efficiency with its spiking mode of operation.

4.2 Dataset

This research assumes efficient heartbeat segmentation from ECG signals and focuses on classification. A preprocessed version of the MIT-BIH dataset, provided by the authors of [26], is selected for classification. This dataset has 109,446 segmented heartbeats, 187 samples each, captured from lead II of an ECG with sampling frequency 125 Hz. The dataset maps heartbeats in the MIT-BIH dataset to five superclasses, shown in Fig. 1: N

(normal), SVEB (supraventricular ectopic beat), VEB (ventricular ectopic beat), F (fusion beat), and Q (unknown). These classes have imbalanced proportions: N (82.77%), SVEB (2.54%), VEB (6.61%), F (0.73%), and Q (7.35%). The authors of [26] provide training and test sets in an equivalent 80/20 split across classes. The training set is further partitioned into a 90/10 split across classes for a validation set.

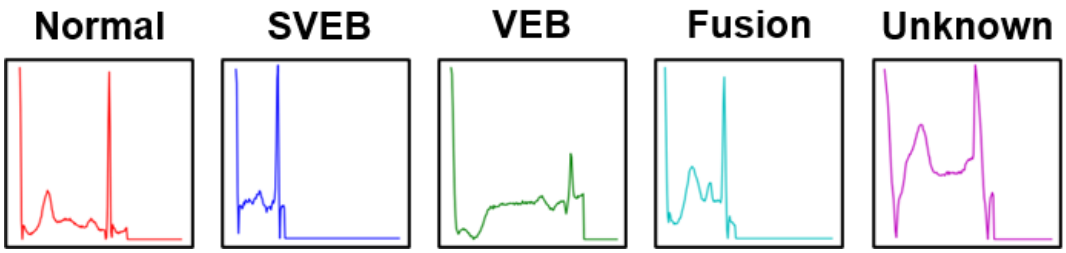

Figure 1: Example Heartbeats From Preprocessed MIT-BIH Dataset [26].

4.3 Artificial Neural Network Design

An ANN must first be trained before using *SNN-Toolbox* to create an SNN. Given the use case of heartbeat classification from ECG signals, MLP and 1D-CNN classifiers are deemed appropriate ANNs for study. However, with past benchmarking of Loihi versus edge neural network devices focusing on MLPs [23], this research focuses on achieving successful conversion with a 1D-CNN and providing novel benchmarks of that architecture. Nonetheless, an MLP is analyzed in the exploration of tuning strategies as well. The MLP architecture of interest is displayed in Table 1. The 1D-CNN architecture of interest is displayed in Table 2. Both models are sufficient for high ANN classification accuracy, while also not having too many layers to significantly impact *SNN-Toolbox* conversion, which suffers from rate approximation errors that accumulate with increasing network depth [4].

Table 1: Baseline MLP Architecture

Layer Type	**Activation Function**	**Output Shape**	**Parameter Count**
Input	-	(187, 1)	-
FC	ReLU	(64)	12,032
FC	ReLU	(64)	4,160
FC	ReLU	(64)	4,160
FC	ReLU	(64)	4,160
FC	ReLU	(64)	4,160
FC	Softmax	(5)	325

Table 2: Baseline 1D-CNN Architecture

Layer Type	**Activation Function**	**Output Shape**	**Filter-Kernel-Stride Configuration**	**Parameter Count**
Input	-	(187, 1)	-	-
Conv1D	ReLU	(92, 8)	(8, 5, 2)	48
Conv1D	ReLU	(44, 16)	(16, 5, 2)	656
Conv1D	ReLU	(20, 32)	(32, 5, 2)	2,592
Flatten	-	(640)	-	-
FC	ReLU	(32)	-	20,512
FC	Softmax	(5)	-	165

Using TensorFlow 2.2.0, each model is trained until there are 10 epochs in which the validation accuracy has not improved. The training configuration consists of a categorical cross-entropy loss function, a batch size of 32, and the Adam optimizer with learning rate 0.001. The learning rate is reduced by a factor of 10 for every 5 epochs in which the validation accuracy does not change.

4.4 Spiking Neural Network Design, Tuning, and Optimization

After training, each ANN is converted to an architecturally identical SNN with *SNN-Toolbox*. The ANN and SNN have the same layers and number of neurons, but the SNN operates with a spiking neuron model. The spiking neuron model requires key hyperparameters to be specified and tuned. One such parameter is the number of timesteps n, which determines how long to present a sample as bias current to the first layer of the SNN. The parameter n fundamentally dictates the number of computational operations per sample, which in turn affects test accuracy and latency. The optimal value of n is captured through exploration of this accuracy-latency tradeoff.

Another parameter is the neuron reset mechanism, which determines how neuron voltages reset after reaching a specified voltage threshold. The neuron mechanism can be a soft reset, which means that at the time a neuron's voltage moves past the threshold, the membrane potential is reset to the current membrane potential minus the voltage threshold. The neuron reset mechanism can also be a hard reset, which causes the membrane potential to be reset to zero after a voltage threshold is reached. Prior research has shown soft reset to achieve better accuracy on MNIST and CIFAR-10 with a 2D-CNN, but also noted that hard reset can be computationally less expensive, albeit at a potential reduction in accuracy [25]. Henceforth, this research found it prudent to extend upon that work by investigating the accuracy effects of the neuron reset mechanism.

Both the accuracy-latency tradeoff and the neuron reset mechanism are explored for the networks in Table 1 and Table 2, with effects demonstrated on final SNN accuracy. Another parameter of note, not varied but specified, is the set of data samples needed for parameter normalization. A subset corresponding to 10% of the training set with equivalent class distribution is used as this representative subset.

Conversion is not guaranteed to generate high accuracy, as poor conversion may result from outliers in the weight, bias, or activity distributions of the ANN [4]. Additionally, the limited fixed-point precision of Loihi (INT8 in this case) constrains the conversion process further. This research considers ANN designs that potentially overcome these challenges. For one, an ANN is designed with bias values that are restricted to be zero during training to prevent large bias values from impacting conversion, as performed in [27]. Second, an ANN with regularization in the form of dropout between the final convolutional layer and first fully connected layer is designed to lower the magnitude of weight and bias values. These designs are trained, converted, and compared to the baseline spiking 1D-CNN in terms of ANN versus SNN accuracy performance.

4.5 Model Evaluation

Evaluation of the 1D-CNN is performed with AAMI recommended metrics of precision (positive predictivity), recall (sensitivity), and false positive rate (FPR). In addition, both MLP and 1D-CNN models are evaluated with overall accuracy and a concise macro-averaged F1 score metric for comparing ANNs to SNNs. These metrics can be expressed for each class i (across N classes) in terms of multiclass true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), as shown in Equations 4-1, 4-2, 4-3, 4-4, and 4-5.

$$Precision = \frac{TP}{TP + FP} \tag{4-1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4-2}$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN} \tag{4-3}$$

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4-4}$$

$$Macro\ F1\ Score = \frac{\Sigma(F1\ Score)_i}{N} \tag{4-5}$$

4.6 Spiking Neural Networks on Loihi

Once tuned, the spiking 1D-CNN is mapped to Loihi through the Intel NxSDK software, which provides the NxTF interface to partition SNNs across the neuron cores of Loihi. When running inference on Loihi, a sample is sent from a host CPU through Loihi's x86 cores to Loihi's neuron cores. A sample is run for n timesteps. Each timestep includes spiking time for neurons to send spikes and update state, and management time for SNIPs to run on the x86 cores to interact with neuron cores. The SNIPs needed for *SNN-Toolbox* are input injection from the x86 cores to the neuron cores, classification readout from the neuron cores to the x86 cores, and reset of neuron core state between samples. After each sample, results are read from the x86 cores to the host.

4.7 Performance Benchmarking

The spiking 1D-CNN on Loihi is compared to an architecturally identical ANN on CPU and edge neural network devices. The CPU and host for each edge device is an Intel Core i7-6700 CPU @ 3.40GHz with 16 GB RAM. Kapoho Bay, with two chips, is used as an edge, USB form factor of Loihi. The Intel Neural Compute Stick 2 (NCS2) and Google Coral Edge TPU (Edge TPU) are selected as state-of-the-art devices for neural network inference at the edge [28]. The NCS2 uses the Intel Movidius Myriad X Vision Processing Unit (VPU), containing 16 programmable SHAVE cores and a neural compute engine to accelerate inference. The Coral USB Accelerator is built as an edge version of Google's Tensor Processing Unit [29], which leverages systolic arrays to accelerate common neural network operations.

All devices interface from a TensorFlow 2.2.0 script in Python 3.5, but use different frameworks for mapping: Intel NxSDK 0.9.9 and *SNN-Toolbox* 0.5.0 for Loihi, OpenVINO 2021.1.110 for the NCS2, and TFLite (TensorFlow 2.2.0) for the Edge TPU. Model quantization also differs with INT8 for Loihi and the Edge TPU and FP16 for the NCS2. Quantized accuracies for the NCS2 and Edge TPU are found to be equivalent to original accuracy.

Evaluative metrics include latency, power (idle, running, and dynamic), dynamic energy per inference, and EDP. Calculations focus on dynamic energy, the main quantity that Loihi targets. Static energy is noted to be important as well, but is not a focus of this study as Kapoho Bay, a research device, is not made with production-grade fabrication like the other devices. Dynamic energy per inference is calculated as the product of dynamic power and latency (Equation 4-6) and EDP as the product of dynamic energy per inference and latency (Equation 4-7).

$$Dynamic\ Energy\ Per\ Inference = Dynamic\ Power * Inference\ Latency \tag{4-6}$$

$$EDP = Dynamic\ Energy\ Per\ Inference\ *\ Inference\ Latency \tag{4-7}$$

EDP is used in particular to account for the tradeoff between energy and delay in CMOS circuitry. These metrics are calculated for unbatched inference to match this online use case.

Latency is measured as the average inference time over a loop of 1,000 samples. Measurements of latency are further averaged over 10 trials for each device. Power measurement across devices, on the other hand, is noted to be challenging due to differences in available measurement methodologies. Nonetheless, this research aims to provide high quality estimates of power through direct measurement during inference, with inspiration in methodology from [23]. Across all devices, idle power is measured from boot. For all devices except Loihi, average running power is calculated during a loop of inference. Dynamic power is then calculated as average running power subtracted by average idle power. The CPU uses the s-tui software to measure package power, recording values every 2 seconds for 5 minutes, while the NCS2 and Edge TPU use inline USB and USB-C multimeters, respectively, with power recorded every 10 seconds for 2 minutes. These measurements are further averaged for the CPU over 5 trials and for the NCS2 and Edge TPU over 3 trials. Given the standard deviations across trials, these numbers of trials are deemed sufficient for benchmarking. For the edge hardware, the USB and USB-C meter precisions are two-decimal and three-decimal, respectively. Due to their inline nature, measurements capture total system power draw, including USB I/O.

NxSDK probes are used to measure average idle, dynamic, and running power of a Loihi chip on Kapoho Bay. These probes are also used to measure dynamic power of the neuron and x86 cores. The probing frequency on Kapoho Bay makes it difficult to accurately capture neuron core power at low timestep counts, so samples are run for 8,192 timesteps to collect an adequate number of readings. Readings are collected for five samples which is sufficient to measure average power without overwhelming embedded memory. Power is further averaged across 10 trials to make the estimates more robust. As these power estimates are averages over entire inferences, it is not possible to isolate power for spiking, management, and readout stages. In energy calculations, x86 and neuron core power are therefore assumed to be always active during an inference, leading to a conservative estimate of total dynamic inference energy.

5.0 Results

The results of this research are presented as three topics. First, an exploration of MLP and 1D-CNN classifiers in *SNN-Toolbox* is conducted. This exploration covers accuracy-latency tradeoffs, neuron reset mechanisms, and weight and bias considerations. Next, the optimized spiking 1D-CNN model is evaluated in terms of AAMI metrics versus the original baseline ANN. Lastly, the spiking 1D-CNN on Loihi is compared to the ANN on CPU, Intel NCS2, and Edge TPU devices in terms of runtime and energy metrics. Measurements and calculations are represented at the precision at which measurements are made.

5.1 Spiking Neural Network Design, Tuning, and Optimization

This section describes the exploration of ANN-to-SNN conversion with the *SNN-Toolbox* framework. The insights presented in this study inspire the 1D-CNN architecture optimized for benchmark analysis. All accuracy results presented are final evaluations on the test set.

5.1.1 Accuracy-Latency Tradeoff

An example of the accuracy-latency tradeoff of SNNs is shown for the baseline 1D-CNN with a soft neuron reset mechanism in Fig. 2. Included in this chart are the baseline ANN accuracy and macro-averaged F1 score achieved over the test set in green, and maximum SNN accuracy and macro-averaged F1 score achieved over the test set for n up to 80 timesteps in black. This tradeoff shows how SNN accuracy can increase with more timesteps. In particular, the tradeoff in Fig. 2 shows that the network begins to plateau in accuracy around 32 timesteps and reaches its maximum test set accuracy (over the simulation duration) of 95.8% at 64 timesteps. Consideration of this tradeoff is important in this performance-oriented study, as the goal is to create a 1D-CNN with high accuracy and low latency.

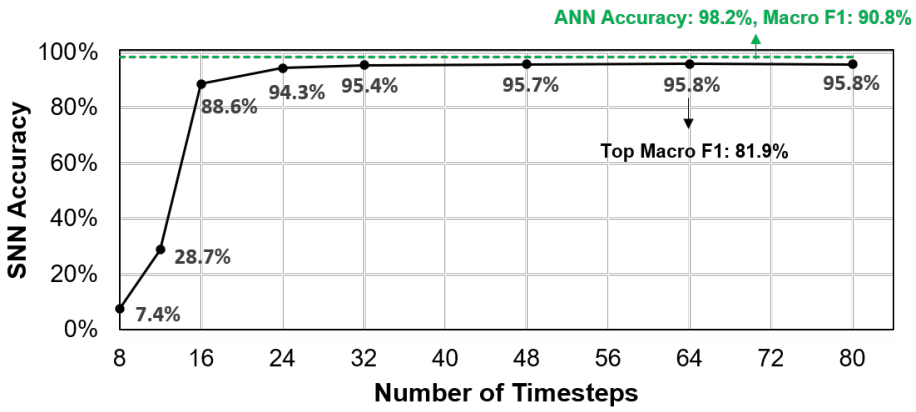

Figure 2: Accuracy-Latency Tradeoff for Spiking 1D-CNN, Evaluated on Test Set, Soft Neuron Reset Mechanism.

5.1.2 Neuron Reset Mechanism

The soft reset accuracy-latency tradeoff for the baseline 1D-CNN has been shown in Fig. 2. With regards to the hard reset, it is found that hard reset timesteps are shorter in duration than soft reset timesteps, but more are needed to improve accuracy. Therefore, the hard reset timestep count ranges are higher to see where the network may plateau. The hard reset accuracy-latency tradeoff over the test set for the baseline 1D-CNN for n up to 1,024 timesteps is exhibited in Fig. 3. The hard reset 1D-CNN is shown to have an unstable progression in accuracy over time, demonstrating poor conversion. The soft reset 1D-CNN is observed to be closer to non-spiking performance, but to still have small differences in macro-averaged F1 score and accuracy versus the ANN baseline (8.9% and 2.4%, respectively).

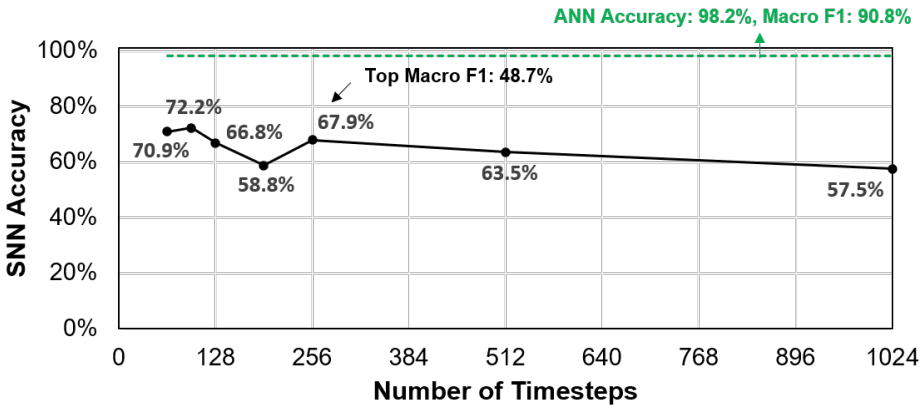

Figure 3: Accuracy-Latency Tradeoff for Spiking 1D-CNN, Evaluated on Test Set, Hard Neuron Reset Mechanism.

This investigation is similarly conducted for the baseline MLP. The soft reset accuracy-latency tradeoff for the MLP is shown in Fig. 4 (up to 96 timesteps), and the hard reset accuracy-latency tradeoff for the MLP is shown in Fig. 5 (up to 1,024 timesteps). In comparison to an accuracy of 98.1% and a macro-averaged F1 score of 90.3% for the ANN, the soft reset mechanism achieves almost exactly non-spiking performance with an accuracy of 98.1% and macro-averaged F1 score of 90.2%. Meanwhile, while the hard reset accuracy-latency tradeoff for the MLP shows a clear increase in accuracy, unlike the 1D-CNN, the MLP achieves only an accuracy of 92.4% and macro-averaged F1 score of 73.8%.

In a situation where the accuracy values of the classifiers are close between the soft and hard reset mechanisms, a performance investigation of the hard reset could be warranted. However, this situation is found to not be the case during the spiking 1D-CNN design process, resulting in benchmarking focus on soft reset.

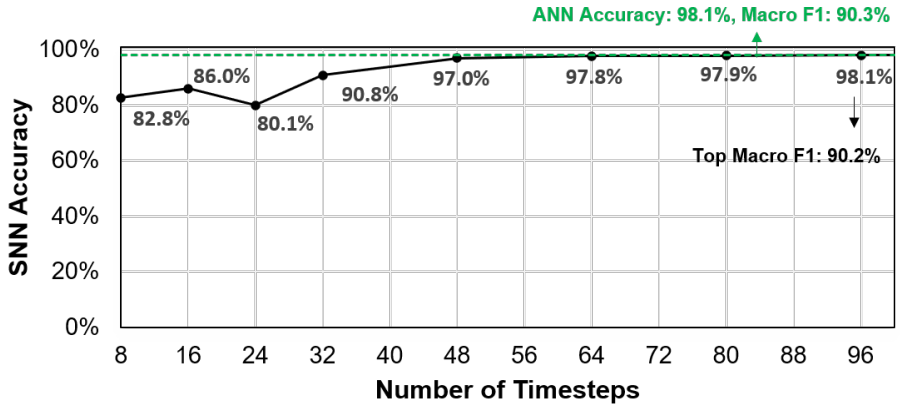

Figure 4: Accuracy-Latency Tradeoff for Spiking MLP, Evaluated on Test Set, Soft Neuron Reset Mechanism.

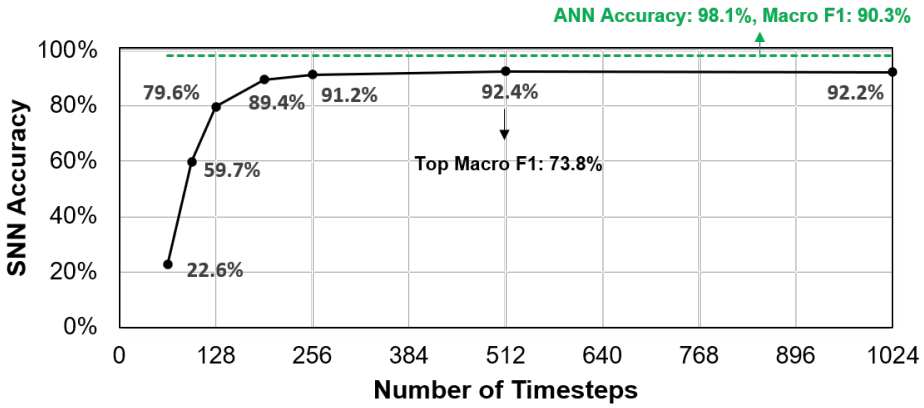

Figure 5: Accuracy-Latency Tradeoff for Spiking MLP, Evaluated on Test Set, Hard Neuron Reset Mechanism.

5.1.3 Weight and Bias Exploration

As referred to in Fig. 2, the maximum macro-averaged F1 score for the soft reset 1D-CNN is 81.9%. This value represents a dropoff from the 90.8% macro-averaged F1 score of the baseline ANN architecture. This research tests regularization and bias restriction

as potential SNN design considerations for achieving higher conversion success. In these experiments, new ANNs with bias values restricted to be zero and with regularization are trained and converted. In particular, regularization is performed by adding a dropout layer with drop rate 25% between the last convolutional layer and first fully connected layer in the 1D-CNN. A comparison between ANN and maximum SNN macro-averaged F1 scores (up to 512 timesteps) over the test set for these various strategies is shown in Fig. 6. Overall, it is shown the restriction of bias values and inclusion of regularization, in isolation and together, demonstrate better conversion in reducing the ANN-SNN gap in macro-averaged F1 score. The model with just dropout achieves the best macro-averaged F1 score of 87.9%.

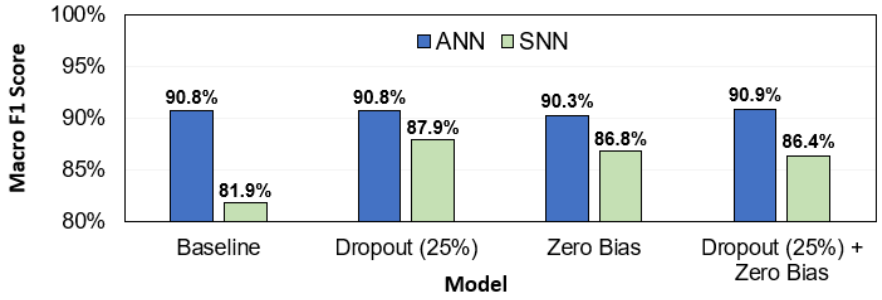

Figure 6: Effects of Regularization and Bias Restriction on Spiking 1D-CNN Conversion, Represented By Gap Between ANN and SNN Macro-Averaged F1 Score on Test Set.

A 1D-CNN with a soft reset and dropout is used for benchmarking on Loihi. The accuracy-latency tradeoff of this model over the test set is shown in Fig. 7. As a final consideration regarding the number of timesteps, since maximum accuracy and macro-averaged F1 score are first achieved at 64 timesteps, this value is chosen as n. Therefore, the 1D-CNN can be run at the minimum latency needed to achieve its top accuracy.

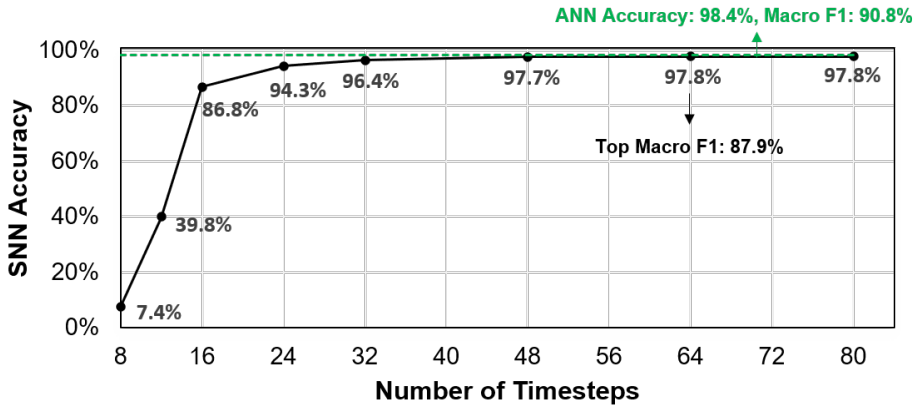

Figure 7: Accuracy-Latency Tradeoff for Spiking 1D-CNN with Dropout, Evaluated on Test Set, Soft Neuron Reset Mechanism.

5.2 Spiking 1D-Convolutional Neural Network Evaluation

Evaluation of the ANN versus the spiking 1D-CNN with a soft reset and dropout at 64 timesteps is presented in Fig. 8, which shows confusion matrices, Table 3, which shows per-class metrics of recall, precision, and false positive rate, and Table 4, which shows total accuracy and macro-averaged F1 score. As with past literature [30], the prediction of a heartbeat as class VEB when it is class F or class Q does not count as a false positive for class VEB. Similarly, a prediction of class SVEB from class Q is not considered a false positive for class SVEB. In general, the ANN achieves better recall, precision, and false positive rate across classes, but the SNN is close in metrics especially for classes N, VEB, and Q. The SNN recall values for classes SVEB and F trail the ANN recall values more significantly. Of note, these classes have the lowest proportions of training samples. Overall, it is found that the accuracy and macro-averaged F1 score of the SNN are 0.6% and 2.9% lower than those of the ANN.

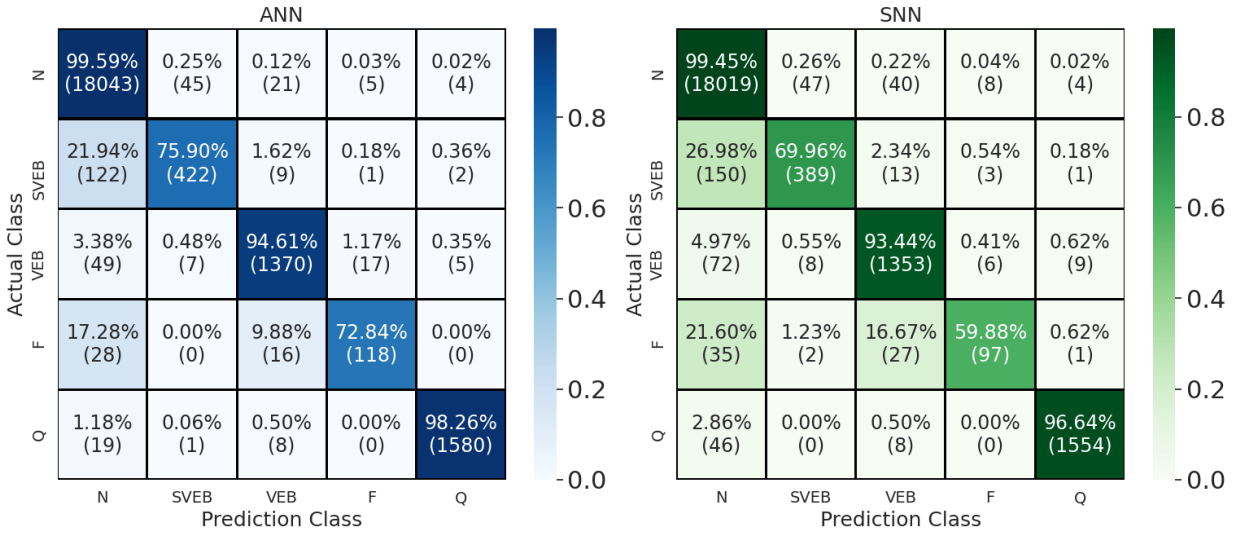

Figure 8: Confusion Matrix Heatmaps Normalized By Class Size for ANN (blue) and SNN (green).

Table 3: Per-Class Metric Comparison

Class	ANN			SNN		
	Recall (%)	***Precision*** (%)	***FPR*** (%)	***Recall*** (%)	***Precision*** (%)	***FPR*** (%)
N	99.6	98.8	5.78	99.5	98.3	8.03
SVEB	75.9	89.0	0.244	70.0	87.2	0.267
VEB	94.6	97.9	0.147	93.4	96.2	0.260
F	72.8	83.7	0.106	59.9	85.1	0.0782
Q	98.3	99.3	0.0542	96.6	99.0	0.0739

Table 4: Overall Metric Comparison

Metric	ANN	SNN
Accuracy (%)	98.4	97.8
Macro-Averaged F1 Score (%)	90.8	87.9

5.3 Performance Profiling and Benchmarking

This section outlines performance profiling in terms of Loihi-specific analysis, followed by latency, power, and energy metric comparisons across devices. For a complete tabular view of all measurements and calculations, please refer to Tables 5, 6, and 7 in the Appendix.

5.3.1 Loihi Performance

First, the resource consumption of the SNN on Loihi (Kapoho Bay) is analyzed. The SNN occupies 9 out of a total 128 neuron cores. A certain number of neuron cores must also be active during an inference as Loihi has a barrier mechanism to ensure spiking neurons are synchronized. A total of 72 neuron cores are active due to these constraints.

Average latency for the SNN on Loihi is captured over 10 trials of averages over 1,000 samples, leading to a value of 7.222 ms. Through NxSDK profiling, an additional breakdown of latency by components is captured. The three components are displayed in Fig. 9. The computational component is spiking, where neurons update state and transmit spikes. This component is shown to be the smallest contributor to latency at 0.798 ms. Another key component for Loihi is management, which involves the use of x86 cores to manage neuron cores, inject input, and read out classifications. This management component is the largest contributor to Loihi runtime at 3.464 ms. Lastly, I/O between the x86 cores and host CPU is the third component to runtime. This component is the second largest at 2.960 ms. These results show that Loihi runtime is bottlenecked by x86 core-based management and x86 core-to-host I/O rather than by spiking computation.

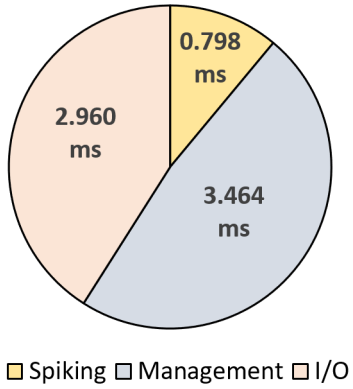

Figure 9: Latency Breakdown for Spiking 1D-CNN on Kapoho Bay.

Average dynamic power for the SNN on a Loihi chip is captured across 10 trials of averages over 5 samples. Fig. 10 shows the breakdown of dynamic power by neuron core power and x86 core power, with error bars included to represent the standard deviation of measurements across trials. Overall, the SNN is found to use 58 mW of dynamic power on average, with 38 mW from the neuron cores and 20 mW from the x86 cores.

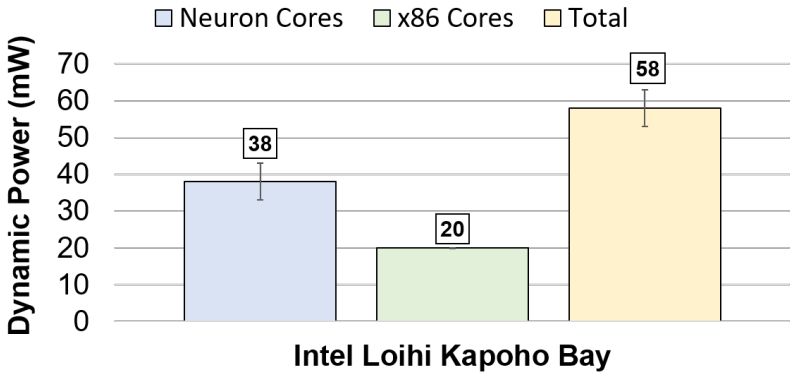

Figure 10: Dynamic Power Breakdown for Spiking 1D-CNN on Kapoho Bay.

5.3.2 Latency Benchmarks

Given the use case of online heartbeat classification, latency is measured across devices in terms of unbatched inference time, where one heartbeat is classified at a time. The average unbatched latency captured over 10 trials of 1,000 sample runs, with associated error bars corresponding to the standard deviation across trials, is displayed for each device in Fig. 11. The spiking 1D-CNN latency on Loihi is observed as the highest at 7.222 ms. The baseline CPU has the next highest latency at 2.455 ms. The NCS2 carries the third highest latency at 1.787 ms. The best-in-class latency is that of the Edge TPU at 0.204 ms.

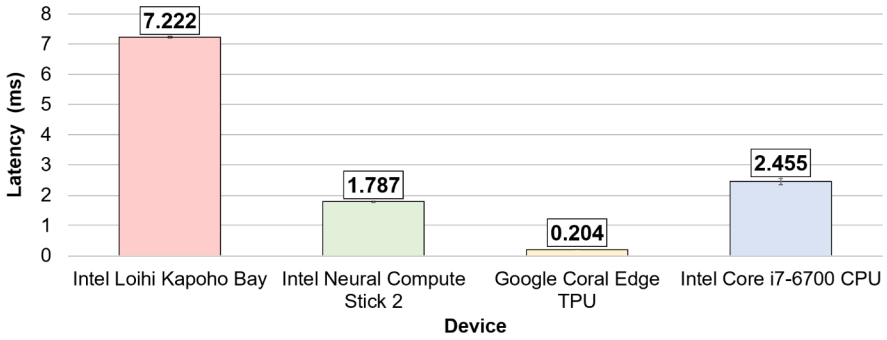

Figure 11: Latency Across Devices.

5.3.3 Power Benchmarks

Power is also measured for comparison across devices. The CPU used in this study is desktop-class, which consequently has higher power magnitude than the other edge architectures. Therefore, the comparison between Loihi and the CPU is demonstrated on a logarithmic plot to show order of magnitude differences in power. A comparison of the average idle, running, and dynamic power for an Intel Core i7-6700 CPU (over 5 trials of averages over 5 minutes) versus Loihi (over 10 trials of averages over 5 samples), with standard deviation error bars for any measurements, is shown in Fig. 12. The Loihi chip on Kapoho Bay operates at significantly less idle and running power than the CPU. The dynamic power difference is significant as well at 240×. These benchmarks make sense given the Kapoho

Bay being an edge form factor of Loihi and the CPU being a desktop-class computer. In general, this CPU benchmark serves as a reference of Loihi versus a standard von Neumann baseline.

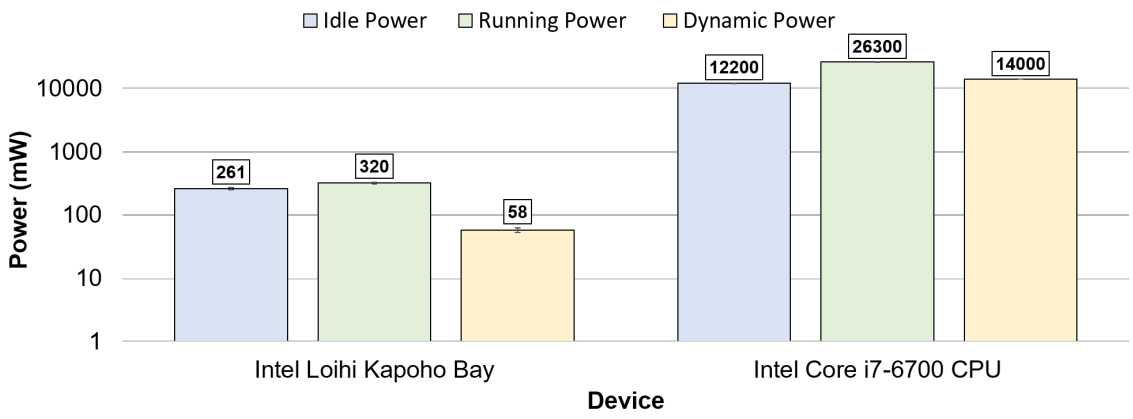

Figure 12: Loihi vs. CPU Power.

To provide a more rigorous comparison, the power of Loihi is compared to state-of-the-art edge neural network accelerators. Fig. 13 shows the average idle, running, and dynamic power measurements for Loihi (over 10 trials of averages over 5 samples), the NCS2 (over 3 trials of averages over 2 minutes), and the Edge TPU (over 3 trials of averages over 2 minutes), with associated standard deviations across trials for any measurements. The power measurements collected from the NCS2 and TPU represent the entire system power (chip and USB I/O interface) as they are measured with inline voltage-current meters. This consideration can potentially explain the differences in idle power versus Loihi, as idle power for Loihi does not include its USB interface. Nonetheless, by benchmarking power at idle and running phases, the dynamic power of each platform is able to be estimated and used for energy calculations. It is found that the NCS2 and Edge TPU operate with similar dynamic powers of 620 mW and 659 mW, respectively, for the 1D-CNN. Loihi demonstrates the lowest dynamic power across all platforms, needing only 58 mW to run the 1D-CNN.

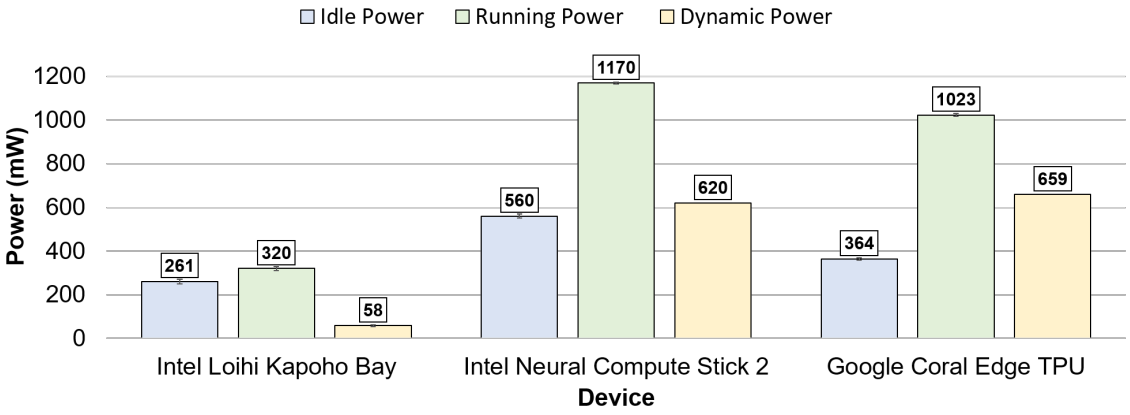

Figure 13: Loihi vs. NCS2 and Edge TPU Power.

5.3.4 Dynamic Energy Per Inference and Energy-Delay Product Benchmarks

A comparison in dynamic energy per inference for Loihi versus the CPU is shown on a logarithmic plot in Fig. 14. Loihi is estimated to use 82× less dynamic energy per inference than the CPU. A comparison in dynamic EDP, which involves a further multiplication by average latency, is shown for the CPU versus Loihi in Fig. 15. The gap in EDP between devices is smaller than the gap in dynamic energy per inference because the latency of Loihi is higher than that of the CPU, and the EDP metric weighs this quantity more. Overall, Loihi is estimated to provide a 28× improvement in EDP versus the baseline CPU.

Loihi is similarly compared to the edge neural network hardware. Fig. 16 shows the differences in dynamic energy per inference between all of the edge devices. The Edge TPU is estimated to use the smallest amount of dynamic energy per inference at 0.134 mJ. A Loihi chip is estimated to use 0.42 mJ, which is 2.6× lower than that of the NCS2. Fig. 17 further shows the EDP across edge devices. Loihi is estimated to have the highest EDP of 3.0 μJs due to its high latency. The EDP of the NCS2, with a value of 2.0 μJs, is 1.5× lower than that of Loihi. Finally, the Edge TPU carries the lowest EDP of 0.0274 μJs. This low EDP of the Edge TPU can be attributed to its much lower latency when compared to the other two devices.

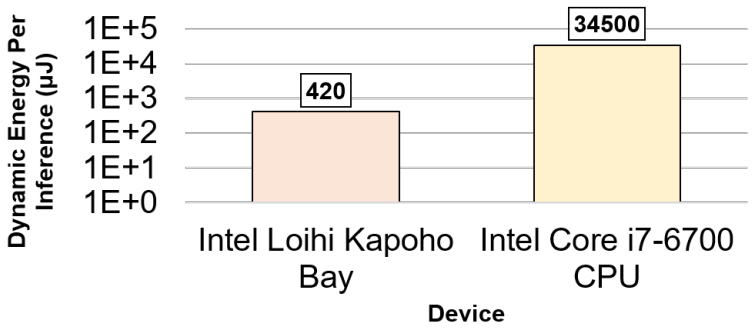

Figure 14: Loihi vs. CPU Dynamic Energy Per Inference.

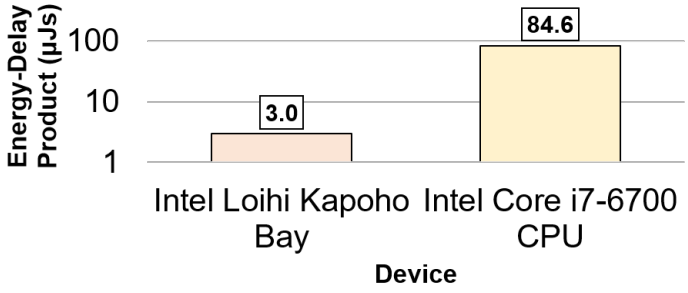

Figure 15: Loihi vs. CPU Energy-Delay Product.

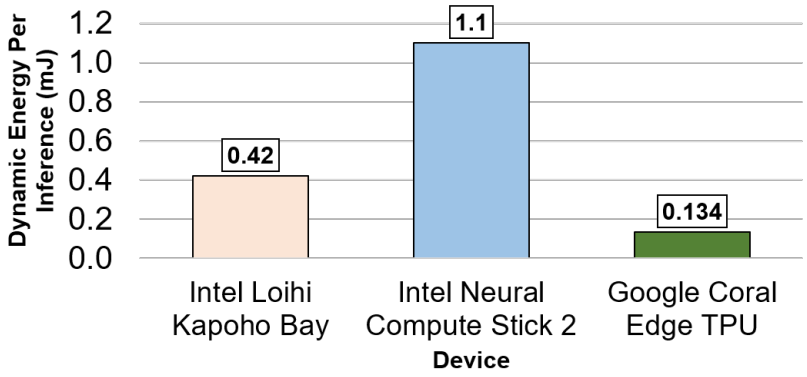

Figure 16: Loihi vs. NCS2 and Edge TPU Dynamic Energy Per Inference.

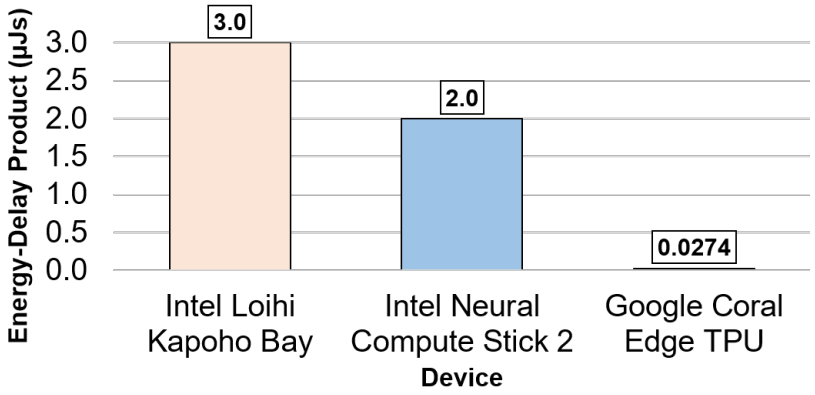

Figure 17: Loihi vs. NCS2 and Edge TPU Energy-Delay Product

6.0 Discussion

This section includes further analysis of the results. The primary insights from the ANN-to-SNN conversion exploration are discussed. Additionally, discussion regarding implications of the performance benchmarking is provided.

6.1 Artificial-to-Spiking Neural Network Conversion

With regards to SNN design through ANN-to-SNN conversion, the accuracy-latency tradeoff is found to be an important consideration for performance. For one, if maximum accuracy is the goal, the designer must be able to find the number of timesteps sufficient to achieve this goal. Moreover, if latency is critical, an SNN allows for accuracy to be traded off for improved latency. This tradeoff can be sufficient for applications where some error is allowed. In this heartbeat classification study, accuracy is collected up to a certain number of timesteps, and the minimum number of timesteps to achieve a top accuracy is selected. This strategy is a way to optimize latency as much as possible without significantly trading off accuracy and thus represents a generalizable method for high-performance SNN design.

The neuron reset mechanism is found to impact accuracy as well, especially in the case of the 1D-CNN. Both the MLP and 1D-CNN are found to favor a soft reset. The hard reset for the 1D-CNN leads to an unstable accuracy-latency progression, where accuracy does not noticeably improve over time. It is hypothesized that this network has low accuracy as a result of information loss from the reset-to-zero operation fanning out through convolution. The limited dynamic range of the SNN, especially given the quantization constraints of Loihi, can also impact this poor conversion. With regards to the MLP, the hard reset and soft reset gap is smaller, and the hard reset MLP shows an accuracy that increases over time. Still, the best macro-averaged F1 score for the hard reset MLP is significantly lower than that of the soft reset MLP. In terms of practical SNN design, the difference in accuracy is deemed too high to justify using the hard reset instead of soft reset for these networks tested.

Lastly, manipulation of neural network weights and biases is shown to affect conversion performance. An SNN with bias values restricted to be zero results in improved conversion versus the baseline. Moreover, regularization in the form of dropout results in a 1D-CNN macro-averaged F1 score closer to non-spiking performance. This improved conversion success compared to the baseline is hypothesized to be a result of smaller weights and biases and thus fewer outliers in the weight, bias, and activity distributions of the regularized ANN. With fewer outliers, there can be more effective correlation between ANN activation values and SNN firing rates. More investigation should be conducted into how to effectively tune weights and biases for successful conversion with such a methodology.

Overall, ANN-to-SNN conversion proves generally successful for the 1D-CNN tested. An SNN with dropout and soft reset is able to achieve close to non-spiking accuracy and macro-averaged F1 score on Loihi. This SNN is tuned to achieve this accuracy at 64 timesteps. Achieving comparable accuracy to an ANN with such strategies is important for practical neuromorphic computing. Still, SNN accuracy has room for improvement. In particular, the recall for classes SVEB and F are noticeably lower than the ANN. This problem can perhaps be remedied with a more balanced dataset and therefore a better subset of data used for conversion. Nevertheless, while ANN-to-SNN conversion works for the network tested, additional investigation should be conducted in how to convert networks of greater scale and diversity. Extending the capabilities of SNN design methodologies in general will enable neuromorphic computing to be a more competitive paradigm for low-power AI.

6.2 Performance Benchmarking

From a latency perspective, Loihi is not the top choice for an accelerator device. It results in the highest latency for the 1D-CNN across devices at 7.222 ms. Tuning the accuracy-latency tradeoff helps minimize this latency to some degree, but as further demonstrated, the minimized latency is bottlenecked by x86 core-based management and x86 core-to-host I/O. These bottlenecks are not fundamental to the neuromorphic chip itself and can potentially be improved with a different choice of CPU. Improving such architectural constraints can help future neuromorphic chips improve performance.

In contrast to its high latency, the dynamic power of Loihi for the 1D-CNN is found to be the lowest across devices. This low dynamic power can be attributed to the neuromorphic mode of computation on Loihi. Overall, having a low dynamic power is the characteristic that allows Loihi to be competitive versus the CPU and NCS2 in terms of dynamic energy calculations.

In consideration of energy efficiency, Loihi is estimated to result in a significantly lower EDP versus the CPU. Loihi achieves this EDP with heartbeat classification being an unbatched use case. A batched use case is likely to see better CPU performance, as CPUs can parallelize batched neural network inferences, unlike Loihi. Moreover, the lower dynamic power of Loihi is found to not outweigh the lower latencies of the NCS2 and the TPU, leading to a higher EDP for Loihi. Loihi is found to be closer in EDP to the NCS2 than the Edge TPU due to a smaller difference in latency. Still, as the NCS2 and Edge TPU estimates capture total system power draw, including USB I/O, the gap in energy dedicated to compute between those devices and Loihi may be greater. In the context of dynamic energy-efficient neural network inference, this use case benefits from a more mature and inference-targeted device like a TPU, rather than from Loihi and this ANN-to-SNN conversion methodology. Further investigation into performance benchmarks of deeper and more diverse models will lead to greater insights as neuromorphic architectures and algorithms continue to develop.

7.0 Future Research

A future route to improve upon the SNN used in this research is to explore more energy-efficient designs. This research involves rate coding, but temporally coded SNNs can potentially provide higher energy efficiency due to sparser spike communication. It will be important to see if temporally coded SNNs can provide as high accuracies and as low latencies as rate-coded SNNs. In general, experimenting with different SNN design methodologies will lead to further latency and energy performance insights.

Another future option is to encode ECG signals as spikes, rather than as per-sample bias currents, and to feed those spikes into an SNN as ECG signals are collected (like performed in [18]). An SNN can then capture temporal information across multiple heartbeats through these spikes, potentially improving accuracy. Latency can also be potentially improved, as a model that captures the recurrence across heartbeats can potentially not need to perform a model state reset between samples.

In the grander context of neuromorphic computing, it will be important to continue to evaluate new architectures as they emerge. Future architectures can potentially overcome some of the limitations shown in this research to perform neural network inference with higher dynamic energy efficiency. Neuromorphic architectures can also be explored for tasks outside of inference. One such example is on-chip learning, which Loihi supports through customizable learning rules. For the use case of heartbeat analysis in particular, an energy-efficient, personalized heartbeat learning system is possible with this capability on Loihi. This idea serves as another future route to investigate and thus evaluate how neuromorphic computing may be able to help health analytics.

8.0 Conclusions

In this study, Loihi is investigated as a neuromorphic platform for heartbeat classification. An exploration of ANN-to-SNN conversion with *SNN-Toolbox* is conducted, and implications of various design considerations for both MLP and 1D-CNN classifiers are analyzed. Due to the fact that SNNs can present a rise in accuracy over time, the ability to trade off accuracy and latency is one such focus for this performance-oriented study. This tradeoff is determined to have optimization implications depending on if an application is more latency-critical or accuracy-critical. Moreover, it is found that the soft neuron reset mechanism provides higher accuracy compared to the hard reset in the cases of both the 1D-CNN and MLP. The use of the soft neuron reset mechanism, along with dropout, ultimately helps to create a spiking 1D-CNN with close to non-spiking accuracy and macro-averaged F1 score.

The emergence of Loihi as an accessible and flexible neuromorphic platform enables competitive benchmarking of this practical SNN design approach. Through analysis of the spiking 1D-CNN on Loihi versus an ANN on CPU, Intel Neural Compute Stick 2, and Google Coral Edge TPU devices, this study evaluates whether an *SNN-Toolbox* model on Loihi is a potential dynamic energy-efficient alternative to other devices. It is found that Loihi provides the lowest dynamic power across devices, which enables the spiking 1D-CNN to result in a lower energy-delay product on Loihi than that of the CPU baseline. However, due to higher latency, the energy-delay product of Loihi is estimated to be higher than the Intel Neural Compute Stick 2 and Google Coral Edge TPU. Overall, the other edge architectures currently offer more compelling and dynamic energy-efficient alternatives for the model tested due to the I/O and x86-core based management bottlenecks exhibited by the Loihi architecture. The improvement of architectures, along with continued study of SNN design, from accuracy, energy, and latency perspectives, can increase the potential of neuromorphic computing in low-energy use cases like heartbeat classification.

Appendix

There are three tables in this section. Table 5 shows the power measurements for each device. All direct measurements in Table 5 include the standard deviation across trials. Average power is measured across 5 trials of averages over 5 minutes for the CPU, 10 trials of averages over 5 samples for Loihi, and 3 trials of averages over 2 minutes for the NCS2 and Edge TPU. Table 6 shows the average latency, computed across 10 trials of averages over 1,000 samples for all devices, along with the standard deviation across trials. Lastly, Table 7 shows the calculated dynamic energy per inference and energy-delay product for each device.

Table 5: Device Power Measurements

Device		**Power (W)**		
		Idle	***Running***	***Dynamic***
Intel Core i7-6700 CPU @ 3.40GHz		12.2 ± 0.06	26.3 ± 0.2	14.0
Intel Loihi Kapoho Bay	x86 Cores	-	-	0.020 ± 5e-5
	Neuron Cores	-	-	0.038 ± 0.005
	Total	0.261 ± 0.01	0.320 ± 0.01	0.058 ± 0.005
Intel Neural Compute Stick 2		0.56 ± 0.01	1.17 ± 0.004	0.62
Google Coral Edge TPU		0.364 ± 0.005	1.023 ± 0.007	0.659

Table 6: Device Latency Measurements

Device	**Inference Latency (ms)**
Intel Core i7-6700 CPU @ 3.40GHz	2.455 ± 0.1
Intel Loihi Kapoho Bay	7.222 ± 0.03
Intel Neural Compute Stick 2	1.787 ± 0.02
Google Coral Edge TPU	0.204 ± 0.002

Table 7: Device Energy Calculations

Device	**Dynamic Energy Per Inference (mJ)**	**Energy-Delay Product (μJs)**
Intel Core i7-6700 CPU @ 3.40GHz	34.5	84.6
Intel Loihi Kapoho Bay	0.42	3.0
Intel Neural Compute Stick 2	1.1	2.0
Google Coral Edge TPU	0.134	0.0274

Bibliography

[1] M. Verhelst and B. Moons, "Embedded deep neural network processing: Algorithmic and processor techniques bring deep learning to IoT and edge devices," *IEEE Solid-State Circuits Magazine*, vol. 9, no. 4, pp. 55–65, 2017.

[2] M. Davies *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[3] H. K. Weir *et al.*, "Peer reviewed: heart disease and cancer deaths—trends and projections in the United States, 1969–2020," *Preventing Chronic Disease*, vol. 13, 2016.

[4] 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.

[5] A. B. De Luna, *Clinical Electrocardiography, Enhanced Edition: A Textbook*. John Wiley & Sons, 2012.

[6] P. Zimetbaum and A. Goldman, "Ambulatory arrhythmia monitoring: choosing the right device," *Circulation*, vol. 122, no. 16, pp. 1629–1636, 2010.

[7] C. Mead, *Analog VLSI and Neural Systems*. Addison-Wesley Longman Publishing Co., Inc., 1989.

[8] L. F. Abbott, "Lapicque's introduction of the integrate-and-fire model neuron (1907)," *Brain Research Bulletin*, vol. 50, no. 5-6, pp. 303–304, 1999.

[9] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science*, vol. 345, no. 6197, pp. 668–673, 2014.

[10] E. Frady *et al.*, "Neuromorphic nearest neighbor search using Intel's Pohoiki Springs," in *Proceedings of the Neuro-Inspired Computational Elements Workshop*, pp. 1–10, 2020.

[11] E. J. S. Luz, W. R. Schwartz, G. Cámara-Chávez, and D. Menotti, "ECG-based heartbeat classification for arrhythmia detection: A survey," *Computer Methods and Programs in Biomedicine*, vol. 127, pp. 144–164, 2016.

[12] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH arrhythmia database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[13] AAMI, *Recommended practice for testing and reporting performance results of ventricular arrhythmia detection algorithms*. Arlington, VA, USA: Association for the Advancement of Medical Instrumentation, 1987.

[14] S. Kiranyaz, T. Ince, and M. Gabbouj, "Real-time patient-specific ECG classification by 1-D convolutional neural networks," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 3, pp. 664–675, 2015.

[15] S. Saadatnejad, M. Oveisi, and M. Hashemi, "LSTM-based ECG classification for continuous monitoring on personal wearable devices," *IEEE Journal of Biomedical and Health Informatics*, vol. 24, no. 2, pp. 515–523, 2019.

[16] N. Wang, J. Zhou, G. Dai, J. Huang, and Y. Xie, "Energy-efficient intelligent ECG monitoring for wearable devices," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 5, pp. 1112–1121, 2019.

[17] F. Corradi *et al.*, "ECG-based heartbeat classification in neuromorphic hardware," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2019.

[18] F. C. Bauer, D. R. Muir, and G. Indiveri, "Real-time ultra-low power ECG anomaly detection using an event-driven neuromorphic processor," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 6, pp. 1575–1582, 2019.

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

[20] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep learning in spiking neural networks," *Neural Networks*, vol. 111, pp. 47–63, 2019.

[21] S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," *Advances in Neural Information Processing Systems*, vol. 31, pp. 1412–1421, 2018.

[22] T. Bekolay *et al.*, "Nengo: a Python tool for building large-scale functional brain models," *Frontiers in Neuroinformatics*, vol. 7, p. 48, 2014.

[23] P. Blouw, X. Choo, E. Hunsberger, and C. Eliasmith, "Benchmarking keyword spotting efficiency on neuromorphic hardware," in *Proceedings of the 7th Annual Neuro-Inspired Computational Elements Workshop*, pp. 1–8, 2019.

[24] E. Ceolini *et al.*, "Hand-gesture recognition based on EMG and event-based camera sensor fusion: A benchmark in neuromorphic computing," *Frontiers in Neuroscience*, vol. 14, 2020.

[25] R. Massa, A. Marchisio, M. Martina, and M. Shafique, "An efficient spiking neural network for recognizing gestures with a DVS camera on the Loihi neuromorphic processor," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9, 2020.

[26] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "ECG heartbeat classification: A deep transferable representation," in *IEEE International Conference on Healthcare Informatics (ICHI)*, pp. 443–444, IEEE, 2018. Dataset URL: https://www.kaggle.com/shayanfazeli/heartbeat.

[27] P. U. Diehl, D. Neil, J. Binas, M. Cook, S.-C. Liu, and M. Pfeiffer, "Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing," in *International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2015.

[28] L. Kljucaric, A. Johnson, and A. D. George, "Architectural analysis of deep learning on edge accelerators," in *IEEE High Performance Extreme Computing Conference (HPEC)*, pp. 1–7, IEEE, 2020.

[29] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pp. 1–12, 2017.

[30] P. De Chazal, M. O'Dwyer, and R. B. Reilly, "Automatic classification of heartbeats using ECG morphology and heartbeat interval features," *IEEE Transactions on Biomedical Engineering*, vol. 51, no. 7, pp. 1196–1206, 2004.