Evaluating Reservoir Spiking Neural Network Configurations for Accurate Battery State-of-Health Prediction

Muhammad Raihaan Kamarudin[1]*, Muhammad Noorazlan Shah Zainudin[2], Mohd Syafiq Mispan[3], and Raihani Mohamed[4]

[1]Fakulti Kecerdasan Buatan dan Keselamatan Siber, Universiti Teknikal Malaysia Melaka, Malaysia; raihaan@utem.edu.my
[2]Fakulti Kecerdasan Buatan dan Keselamatan Siber, Universiti Teknikal Malaysia Melaka, Malaysia; noorazlan@utem.edu.my
[3]Fakulti Teknologi dan Kejuruteraan Elektronik dan Komputer, Universiti Teknikal Malaysia Melaka, Malaysia; syafiq.mispan@utem.edu.my
[4]Fakulti Sains Komputer dan Teknologi Maklumat, Universiti Putra Malaysia, Malaysia; raihanimohamed@upm.edu.my

***Correspondence:** *Muhammad Raihaan Kamarudin, Email: raihaan@utem.edu.my*

ABSTRACT- Accurate prediction of battery State-of-Health (SoH) is crucial for ensuring safety, reliability, and longevity in energy storage systems. This study evaluates the utilization of Reservoir Spiking Neural Networks (RSNN) for battery SoH prediction, focusing on how network configuration affects prediction performance. Various RSNN architectures are analyzed by varying reservoir neuron number, connection density, inhibitory ratio, time step window, and readout network configuration. The findings show that a simple RSNN structure is sufficient to achieve high prediction accuracy, provided that the network is carefully configured to produce diversity in the spike count patterns within the reservoir layer. For the SoH dataset used, the original input features alone do not yield sufficient diversity, highlighting the importance of structural tuning in the reservoir. In the optimized network model, the prediction accuracy achieved a lowest error of 0.029 ± 0.011 RMSE, demonstrating strong accuracy and generalization across different battery datasets, with an energy consumption of approximately 150 nJ. Furthermore, this work provides a framework for implementing RSNN models on embedded platforms, including neuromorphic devices such as Intel Loihi and SoC-FPGA platforms with customized hardware circuits. The results demonstrate the potential of RSNNs as lightweight, efficient, and hardware-friendly models for practical battery SoH prediction.

Keywords: Reservoir Spiking Neural Network, State-of-Health Prediction, Neuromorphic Computing, Battery Management System.

ARTICLE INFORMATION

Author(s): Muhammad Raihaan Kamarudin, Muhammad Noorazlan Shah Zainudin, Mohd Syafiq Mispan, and Raihani Mohamed;

Received: 28/08/2025; **Accepted:** 22/11/2025; **Published:** 15/12/2025;
E- ISSN: 2347-470X;
Paper Id: IJEER 2808A19;
Citation: 10.37391/ijeer.130419
Webpage-link:
https://ijeer.forexjournal.co.in/archive/volume-13/ijeer-130419.html

Publisher's Note: FOREX Publication stays neutral with regard to jurisdictional claims in Published maps and institutional affiliations.

1. INTRODUCTION

Battery technology is one of the most crucial components to the development of electric vehicles (EV), storage of renewable energy and uninterruptable power supply (UPS) systems [1-3]. To manage the battery, most systems rely on Battery Management Systems (BMS) which consist of multiple important modules; one of them is the State of Health (SoH) module [4,5]. The purpose of SoH is to measure the battery's health or lifespan relative to the health state of a new battery. Over time, the battery's health will degrade. The degradation rate depends on many factors mainly because of usage conditions, storage temperature, and charge/discharge rate [1,4]. SoH can easily be measured by using *equation 1* which indicates capacity and power fade as internal resistance increases. Here, C_{now} denotes the maximum allowable discharge capacity, and C_0 denotes the battery's nominal capacity.

$$SoH = \frac{C_{now}}{C_0} \times 100\% \tag{1}$$

Currently, various techniques are available to monitor and estimate the health state of a battery. These include: *(i)* direct measurement, *(ii)* internal state modeling, and *(iii)* data-driven methods [6]. The data-driven technique offers advantages over other approaches, particularly in terms of implementation practicality and real-time adaptability. The main reason is that the data-driven method relies on machine learning techniques to estimate the battery's health state according to historical data [6]. The current trend in data-driven techniques mostly uses deep learning algorithms such as Convolutional Neural Network (CNN) [6-10] and Long Short-Term Memory (LSTM) models [11-14] to produce good estimation of battery SoH. Deep learning techniques rely on the high quantity and quality of the acquired data to produce accurate estimation. However, current battery degradation data are mostly recorded in controlled and ideal environments [15]. In real applications, battery degradation is a complex process in which monitoring and estimating the battery's state in real-time is challenging, as it usually relies on incomplete and low-quality data. Furthermore, the computational burden increases rapidly when the data increase, especially for complex deep-learning models [1,2] that hinder on-situ learning. Recently, a few research works [16,17] have utilized the concept of Spiking Neural Network (SNN) to resolve the limitations posed by deep learning techniques. SNN, also well-known as the 3rd generation of neural networks, is theoretically more power-efficient and promising than current deep learning techniques [18-20]. Therefore, this work explores the suitability of the Reservoir Spiking Neural Network (RSNN) model for predicting battery State of Health (SoH) by evaluating both prediction accuracy and computational cost. The main contributions of this work are summarized as follows:

- A detailed evaluation of RSNN performance in battery SoH prediction is provided, considering different network configurations such as reservoir size, connection density, inhibitory ratio, spike encoding methods, and readout structures.
- It is demonstrated that a simple RSNN structure, when properly configured to generate spike pattern diversity, can achieve high prediction accuracy with low computational cost.

The rest of this article is organized as follows: *Section 2* reviews related work on SoC estimation and spiking neural networks. *Section 3* describes the experimental setup, while *section 4* presents the results and discussion. Finally, *section 5* concludes the study and outlines future research directions.

2. RELATED WORK

2.1. Current Techniques on Estimating Battery's State of Health (SoH)

Battery SoH can be estimated using various techniques. In direct measurement technique, the capacity and the internal resistance is measured within a test lab environment. The coulomb counting method, open-circuit voltage (OCV) method, and internal resistance-based estimating method are all common ways in this technique. These approaches are applicable to a variety of batteries and are simple to execute. However, the accuracy of the estimation is strongly dependent on the measuring procedure. Furthermore, this type of technique is only relevant for off-line SoH measurement in a test lab environment [4].

Meanwhile, internal modeling approaches are techniques that use prior knowledge to represent the internal battery's condition as an electrochemical model. The model parameters, which include the aging characteristics of the lithium battery, are found using the least squares approach or the observer method, and the SoH is estimated using the model parameters. However, selecting the correct model to achieve a trade-off between SoH estimation accuracy and computing cost is difficult, limiting its implementation in practice [1]. In the data-driven method, the idea is to extract available aspects of battery degradation from battery characterisation data and then use machine learning algorithms to build a link between these features and SOH. For instance, the cycle number, incremental capacity, differential voltage, and candidate features in the voltage response under the current pulse test are chosen as features to represent battery degradation. Then, machine learning methods such as Support Vector Machine (SVM), Gaussian Process Regression (GPR), Grey Relational Analysis, and artificial neural networks (ANN) [2,5] are used to learn the nonlinear mapping from characteristics to SoH. Compared with these three techniques, the data-driven approach has advantages in terms of prediction accuracy on real-life situation because it is based on historical data. Additionally, this technique is easier to deploy in real application [2,3].

Recently, there has been significant progress on data driven methods for SoH estimation using Deep Learning techniques. Deep learning techniques for SoH estimation have gained significant interest as they produce higher prediction accuracy compared to other machine learning techniques [2]. For instance, the research work in [6] uses Convolutional Neural Network (CNN) for local features with transformer-based global variables to estimate battery state with higher accuracy. Although previous research has employed various data preprocessing methods to reduce model complexity, these approaches still suffer from high computational demands and are typically optimised for a single type of battery. Similarly, the research work in [10] also utilizes modified CNN with explicit preprocessing techniques to reduce noise on an offline dataset, resulting higher and more robust battery state estimation. However, the model also suffers from a high computational process and was only tested on a limited number of battery offline datasets. In another approach, Long Short-Term Memory (LSTM), also known as a recurrent neural network, has also been widely used in this research area. For example, the research work in [11] uses LSTM to predict battery state of health with high accuracy although there is a specific limitation where the model is designed with only a limited input parameter. This reflects the scalability problem if the same estimator design is to be used on multiple battery cells for a large scale storage system. There are also other techniques [7-9] that combine different neural network models (CNN and LSTM) to achieve higher and more robust estimation accuracy that work with variant battery dataset. However, these models still suffer from computational complexity and are not suitable to be implemented on low-powered edge computing platform.

2.2. Spiking Neural Network as an Alternative Approach

The Spiking Neural Network (SNN), known as the third generation of neural networks, is a promising alternative to address the high computational demands of deep neural networks. Unlike previous generations of ANN, information transfer in SNNs is based on discrete spikes, which significantly reduces power consumption. This is achieved because neurons within the network are only active when they receive a spike carrying information from other connected neurons. Hence, it removes the necessity to continuously update their internal state at every single simulation time-step.

Theoretically, SNN has several advantages as compared to conventional artificial neural network. Firstly, when compared to traditional ANN, SNN has the potential to be more efficient and have lower latency. They can achieve cutting-edge accuracy while reducing fire rates and increasing energy efficiency. Secondly, SNN is event-driven, which means it computes only when there is information to process. This results in extremely energy-efficient processing, particularly when dealing with sparse data from the outside world. In addition, SNN can completely leverage accurate temporal information from event-based sensors, making it perfectly suited for processing spatio-temporal event-based information. This enables more efficient feature computing and increases power efficiency. On top of that, because of their low energy consumption, robustness, and ability to decode in real-time [21], SNN is well-suited for real-time applications such as brain-machine interfaces, automatic driving, and robotics. These benefits place SNN as a promising approach for efficient and real-time processing, especially in applications like in EV or renewable energy storage devices that requiring low power consumption and real-time decision-making based on continuous input streams.

Practically, SNN is already widely used in different research areas for instance in object recognition, optical character recognition, robotic applications, and data forecasting [22-24]. In these applications, SNN has been proved to have a significant impact especially in terms of training speed, energy efficiency and requires less demand on computational resources. However, it comes with specific drawbacks mostly in terms of prediction accuracy in which usually slightly less than conventional machine learning counterparts. This limitation comes from the fact that the current technique usually requires conversion from original input data into spike trains, which is typically lossy and inefficient, contributing to this reduced accuracy. Additionally, the lack of training algorithms that make specific use of the capabilities of spiking neurons, as well as the challenges in designing and analyzing training algorithms for SNN further limits their performance [18]. In BMS research area, the utilization of SNN especially for battery state estimation is still immature. To date, only a single research group has applied SNN to predict battery SoH as reported in [16, 17]. These initial works have proved that SNN gives significant advantages not only enhancing the prediction accuracy but at the same time reducing energy consumption for SoH estimation in BMS system.

3. MATERIALS AND METHODS

3.1. Datasets

In this work, the dataset provided by National Aeronautics and Space Administration (NASA) [25] is used. NASA's dataset includes cycles aging data for pouch-type LCO/graphite 1.5 Ah cells. The information includes current, voltage, charge/discharge capacity and energy, internal resistance, and impedance measurements. Cycling at low C-rates was used to age the cells, followed by calendar aging at various storage temperatures. The dataset is available on Kaggle database that directly can be access from here https://www.kaggle.com/code/rajeevsharma993/battery-health-nasa-dataset which comprises experimental data from four lithium-ion batteries (#5, #6, #7, and #18).

3.2. RSNN Architecture

The RSNN network model consists of 3 important network pool; the input layer, the reservoir network and the readout network. *Figure 1* shows the structure of RSNN.

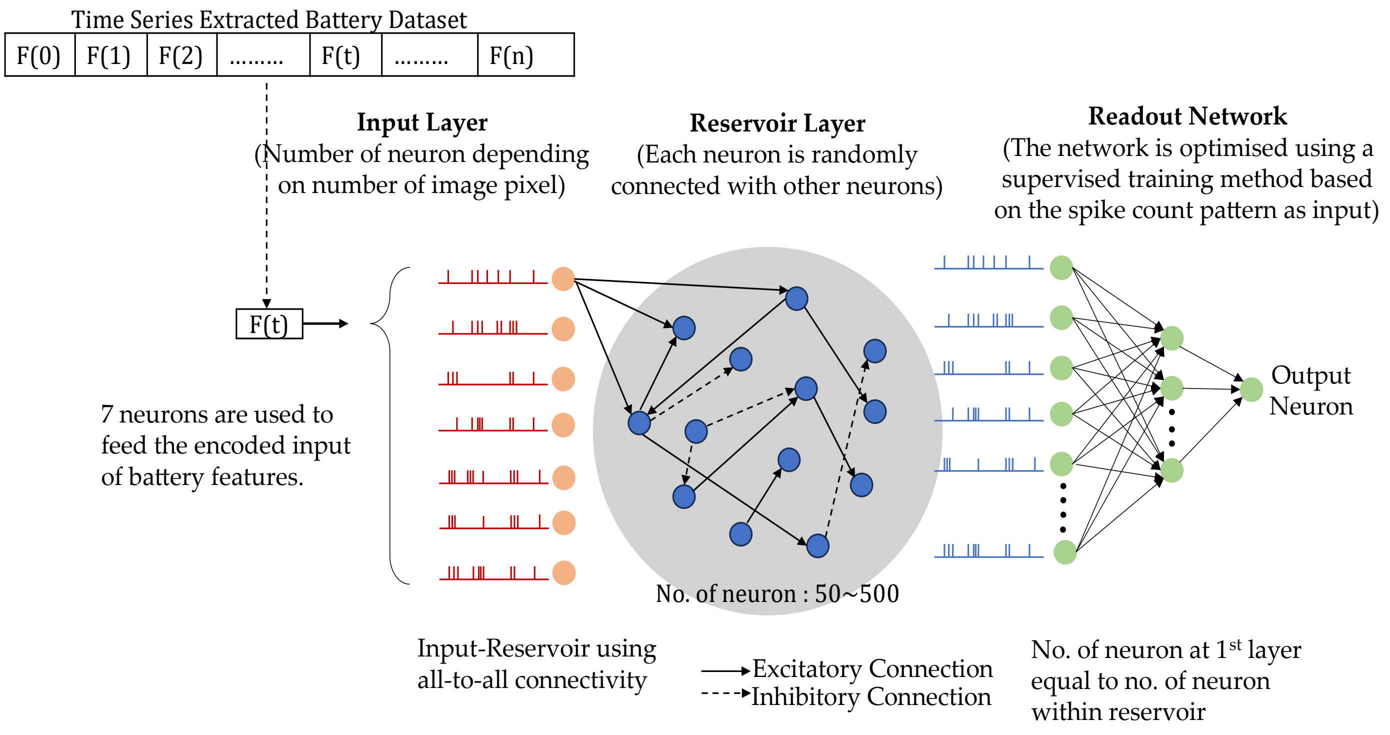
Time Series Extracted Battery Dataset
F(0) | F(1) | F(2) | .... | F(t) | .... | F(n)
Input Layer
(Number of neuron depending on number of image pixel)
Reservoir Layer
(Each neuron is randomly connected with other neurons)
Readout Network
(The network is optimised using a supervised training method based on the spike count pattern as input)
F(t)
7 neurons are used to feed the encoded input of battery features.
Output Neuron
No. of neuron : 50~500
Input-Reservoir using all-to-all connectivity
Excitatory Connection
Inhibitory Connection
No. of neuron at 1st layer equal to no. of neuron within reservoir

Figure 1. Reservoir Spiking Neural Network (RSNN) Architecture. The network model consists of three main parts: the input layer, the reservoir network and the readout network.

3.2.1. Input Layer

The function of input layer basically is to receive encoded data of different features of the battery dataset. The dimensionality of the input layer was determined by the number of features, where each continuous input value was converted into a binary spike train occuring within simulation time steps, *tstep*. The encoded spike trains were then forwarded to the reservoir network via sparse synaptic connections. Each input neuron projected to all reservoir neurons, with each connection assigned a random non-negative synaptic weight.

3.2.2. Reservoir Network

The reservoir network forms the central dynamic core of the RSNN architecture and is composed of interconnected Leaky Integrate-and-Fire (LIF) neurons. This layer nonlinearly projects temporally encoded spike trains from the input layer into a high-dimensional dynamic space. It enables the network to retain short-term memory and process temporal dependencies. Each LIF neuron accumulates postsynaptic potentials from incoming spikes, updating its membrane voltage over time. The reservoir neuron was modeled using a simple LIF formulation, as described in *equation 2*, where a neuron emitted a spike ($s_j(t) = 1$) when its membrane potential $V_j(t)$ exceeded the threshold value, after which it was immediately reset to V_{reset}. In the absence of input spikes, the membrane potential decayed exponentially according to the time constant, τ. Connectivity in the reservoir layer followed a sparse, randomized topology to promote diverse internal dynamics.

$$V_j(t + \Delta t) = V_j(t) + \frac{\Delta t}{\tau}(-V_j(t) + I_j(t)) \tag{2}$$

3.2.3. Readout Network

The readout network constituted the final processing stage of the RSNN, tasked with decoding the high-dimensional spatiotemporal dynamics generated by the reservoir into a scalar output representing the battery's SoH. It began by transforming the reservoir's spike tensor generated over the 100 ms simulation period into a compact feature vector. This was achieved by summing the spiking activity of each reservoir neuron across all time steps, yielding a single activation value per neuron. The resulting vector, with a dimension equal to the number of reservoir neurons, encoded the temporal firing profile in a static form suitable for regression. This spike count vector was then passed into a regression model, implemented as a multiple-layer feedforward neural network. The training process was restricted to the readout layer, while the reservoir weights and connectivity remained static, consistent with the reservoir computing paradigm.

3.3. Data Preprocessing and Encoding

The data preprocessing procedure followed a structured pipeline to prepare the NASA battery dataset for RSNN model training. Key battery parameters which consist of [*capacity, voltage_measured, current_measured, temperature_measured, current_load, voltage_load, time*] were extracted from the discharge cycles. To ensure uniform feature scaling and prevent dominance by any single feature, all variables were normalized using the MinMax scaling technique that standardizes the value range between [0, 1]. Unlike conventional neural network, SNN does not process continuous values. Instead, the input must first be encoded into spike stream For this work, Poisson encoding technique is used to imitate the firing behavior of biological neurons which converted normalized continuous input values into spike trains that randomly occur. Each input value is normalized to fall between [0,1] and multiplied by the predetermined *max_rate*. This determines how many spikes to expect during the simulation window. For example, an input of 0.7 yields a higher expected firing rate than an input of 0.2, leading to more frequent spiking activity. However, the spike timing in Poisson encoding is probabilistic and driven by the average inter-spike interval (ISI). This parameter controls the mean temporal spacing between spikes while allowing for natural variability which results in spike events that are irregularly distributed across time.

3.4. Performance Evaluation

To comprehensively evaluate the performance of the RSNN model for battery SoH prediction, this study considers two key aspects: prediction accuracy and computational cost. Accuracy is assessed using standard regression metrics, namely the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), which are defined in *equation 3* and *equation 4*; respectively.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=0}^{n}(y_i - \hat{y}_i)^2} \tag{3}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4}$$

Here, y_i is the true SoH value, $\hat{y}_i$ is the predicted SoH value, and N is the number of samples. These metrics quantify how closely the RSNN predictions match the actual values. Meanwhile, the computational cost is evaluated using *equation 5*, in which directly measures the total number of spikes generated during inference.

$$Synaptic\ Event = \sum_{t=1}^{tstep}\sum_{n=1}^{N} S_n(t) \tag{5}$$

Here, $S_n\ (t) \in \{0, 1\}$ represents the spike event (1 if a spike occurs, 0 otherwise) of neuron n at time step t, N is the total number of neurons, and *tstep* is the total number of simulation time steps. This spike count serves as a proxy for energy consumption in neuromorphic hardware, since spiking operations typically consume energy only when a spike occurs. While training speed is also observed, it is not used as a primary measure of computational efficiency due to limitations in the simulation environment, which does not fully reflect the real-time performance of neuromorphic devices. Therefore, incorporating spike count offers a more meaningful and hardware-relevant indicator of computational cost in this context.

4. RESULTS AND DISCUSSION

4.1. Effect of Reservoir Size on the Network Performance

One of the critical design parameters in RSNN is the number of neurons in the reservoir pool, which functions as a temporal memory to preserve and transform input patterns. In general, a larger reservoir is expected to capture more complex temporal and spatial dependencies from the input. To investigate this effect, the network model was configured with varying reservoir sizes, ranging from 50 to 500 neurons. The model was then trained and tested using approximately 20% of the dataset selected randomly. A comprehensive comparative evaluation of these configurations is summarized in *table 1.*

Table 1. Performance evaluation of RSNN with varying reservoir neuron numbers

Reservoir Neurons	Training Time (sec)	Training Loss	RMSE	MAE
50	319	0.0031	0.106 ± 0.019	0.090 ± 0.017
100	330	0.0038	0.116 ± 0.018	0.098 ± 0.016
200	493	0.0056	0.115 ± 0.018	0.099 ± 0.014
300	494	0.0064	0.130 ± 0.030	0.112 ± 0.024
400	501	0.0079	0.119 ± 0.009	0.104 ± 0.008
500	509	0.0090	0.143 ± 0.014	0.126 ± 0.011

The results clearly indicate that increasing the reservoir size does not necessarily improve predictive accuracy. In fact, the 50-neuron configuration achieved the best performance, with the lowest loss (0.0031), the smallest error values in terms of RMSE (0.106 ± 0.019) and MAE (0.09 ± 0.017). By contrast, the 500-neuron reservoir yielded the poorest outcomes, including the highest loss (0.009) and the largest error metrics. Training time further emphasized the trade-offs in reservoir sizing. The model with 50 neurons required 319 seconds to train, whereas increasing the reservoir size to 500 neurons required 60% more training time. Despite this, training on a conventional desktop computer with the BindsNET library revealed negligible variation in memory consumption, which remained stable approximately at 4.47 GB, where the model configuration itself required around 0.05 MB. Here, two primary factors likely explain the limited benefits of larger reservoirs. First, the readout stage was relatively simple, consisting of only a single hidden layer with 10 neurons, which restricted its capacity to exploit the richer reservoir dynamics. Second, the use of all-to-all connectivity within the reservoir led to asynchronous and continuous spiking whenever a single neuron was activated. This significantly reduces the diversity of spike patterns which are a critical feature for the readout layer to learn discriminative temporal representations. As a result, increasing the reservoir size not only failed to improve performance but, in some cases, led to performance degradation. This shows that utilizing a small reservoir size is sufficient to achieve good prediction accuracy, which is also suitable for deployment on edge devices with limited computational resources.

4.2. Effect of Connection Density on the Network Performance

Another critical criterion for edge device deployment of the network model is the network topology itself, specifically the connection density between neurons. In this subsection, the connection density between neurons in the reservoir pool is reduced to only 20% from the original all-to-all connection. The connection density between input-reservoir layers remains as all-to-all connection to ensure all input features are effectively transferred to the reservoir pool. *Table 2* summarizes the network model performance and computational cost with different connection densities in the reservoir layer. As indicated, reducing the connections between neurons lowered the training loss and improved prediction accuracy. For instance, when the connection probability between reservoir neurons is set to 0.2, the average training loss reaches 0.0012, which is approximately three times lower than that observed in the all-to-all connectivity configuration. The prediction accuracy also increased by 20%, with average RMSE of 0.093 ± 0.009 and MAE of 0.079 ± 0.008.

These improvements were achieved while significantly reducing the number of synaptic events in the network model. Across multiple simulation runs using different seed numbers, the energy usage at 20% connection density was approximately 80 nJ. This value is about 71% lower than that of full connectivity, assuming a typical energy cost of ~10 pJ per synaptic event [26–28]. This finding is further supported by the spike count (spike rate) distribution across reservoir neurons. As shown in *figure 2*, the spike count pattern became more evenly distributed when the network was configured with lower connection density. At full connectivity (*p=1.0*), the spike distribution is lowest at *H=0.2*. Meanwhile, reducing the connection density with *p=0.2* effectively increases the spike count distribution at *H=0.31*. Here, *H* value is calculated using *equation 5* where, *N* represents number of reservoir neurons, p_i is the probability or the occurrence of specific spike count, and the *log(N)* represents the maximum possible spike rate.

$$Spike\ Distribution, H = -\frac{\sum_{i=1}^{N} p_i \log(p_i)}{\log(N)} \quad (5)$$

This is also supported by mapping the relationship between prediction accuracy (RMSE) and the average spike count per neuron as shown in *figure 3(a)*. The RMSE falls within the range [0.08–0.13] across multiple tests runs for various connection densities. However, lower connection density consistently achieved better prediction accuracy with a lower average spike rate. Focusing on memory footprints to run the network model, reducing the connection density did not significantly reduce the memory. Running the model on a conventional computer requires around 4 GB and 0.05 MB for network configuration. In BindsNET library, synaptic weights are maintained in dense tensor representations where absent connections are stored as zeros, resulting in nearly constant memory allocation regardless of connection density.

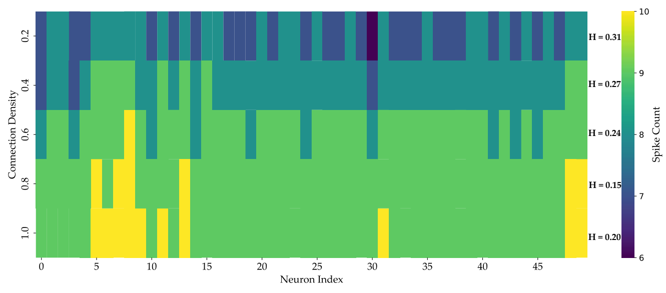

Figure 2. Spike distribution in the reservoir layer for different connection densities. The spike pattern becomes more distributed as the connection density decreases

Consequently, changes in connection density primarily influence spike propagation and computational load, affecting runtime and energy consumption rather than memory usage. However, reducing the reservoir connection density significantly contributes to the memory space savings for the model implementation on edge devices. Assuming that the connection weights are stored as 16-bit fixed-point representation, with a connection probability of 0.2, it only requires 1 kB compared to full connectivity that requires 5 kB. This indicates that a relatively low connection density is sufficient for maintaining prediction accuracy while reducing computational complexity, which is highly advantageous for deployment on edge devices with limited resources.

Table 2. Performance evaluation of RSNN with varying connection densities

Connection Probability	Training Time (sec)	Training Loss	RMSE	MAE	Synapse Number	Synaptic Event
0.2	328	0.0012	0.093 ± 0.009	0.079 ± 0.008	504	8058
0.4	329	0.0018	0.098 ± 0.014	0.083 ± 0.013	1017	12771
0.6	323	0.0026	0.101 ± 0.020	0.085 ± 0.018	1526	18073
0.8	327	0.0029	0.107 ± 0.015	0.090 ± 0.013	2027	23407
1.0	332	0.0031	0.106 ± 0.019	0.090 ± 0.017	2500	28580

4.3. Effect of Window Time on the Network Performance

In spiking neural networks, the input is encoded into spike streams that occur within a specific time window, *tstep*. Theoretically, using a larger value of *tstep* allows the input value to be mapped in more detail. This consequently creates more diversity in the spike rate pattern in the reservoir layer. However, increasing the time window also increases the overall system latency, which affects both the training time and the system response. To analyze the effect of the time window on network model performance, *tstep* is varied from the lowest at 10 ms to the largest at 250 ms. The network topology remains the same, with the connection density of the reservoir layer set at $p = 0.2$. Figure 3(b) shows the relationship between prediction accuracy (RMSE) and the average spike count per neuron. From this relationship, it is clearly indicated that using a low value of *tstep* leads to inaccuracy in the system. For example, at *tstep* = 10 ms, the spike count of each neuron is around 1-2, which is not sufficient to introduce variability in the spike count distribution in the reservoir layer. The detailed spike count distribution is depicted in *figure 4*, where the spike count entropy, *H*, for *tstep* = 10 ms is very low at 0.11. Meanwhile, by using a high value of *tstep*, for instance at 200 ms or 250 ms, the accuracy significantly improves, where the RMSE achieves a low value of approximately 0.072.

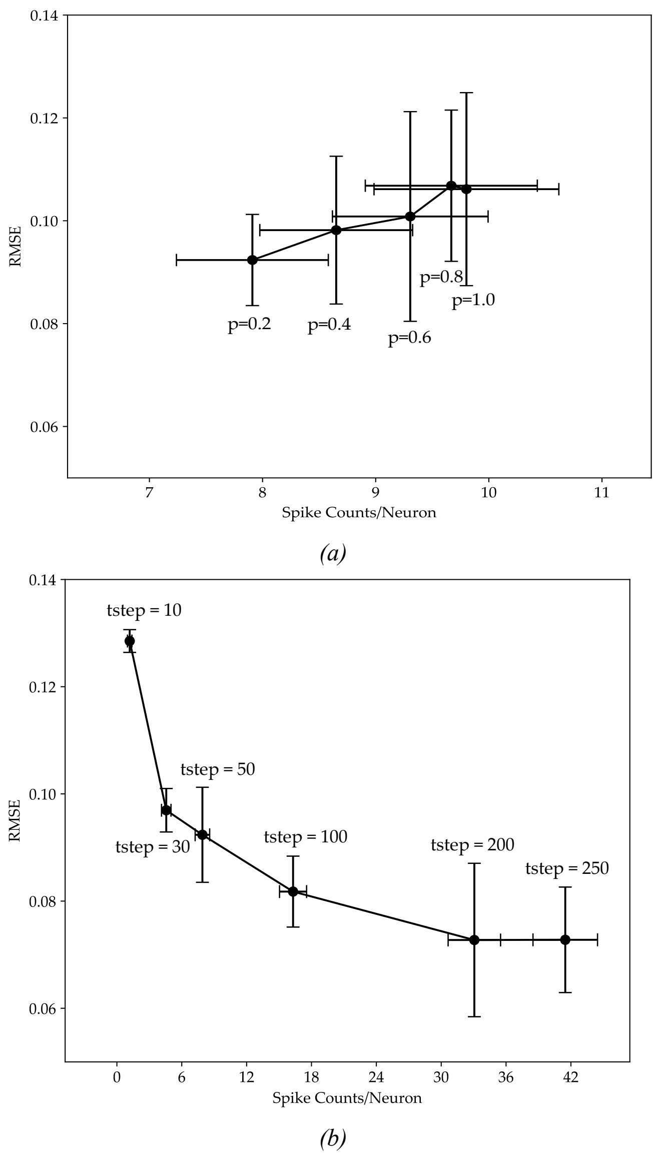

Figure 3. Prediction accuracy (RMSE) versus spike count per neuron: *(a)* with varying connection densities, the RMSE does not change significantly; *(b)* increasing the time step window significantly reduces the RMSE, although it results in a high number of spikes per neuron

The main reason is that the spike count distribution improves greatly as shown in *figure 4*. The entropy value increases at time steps of 200 ms and 250 ms, showing an increase of more than 0.5. Nevertheless, using a high value of *tstep* is not ideal since it increases the overall time latency. Therefore, the next analysis focuses on optimizing the network model using *tstep* varied between 30–100 ms, as the prediction accuracy remains within an acceptable range (RMSE lower than 0.1). This is achieved with less energy consumption (lower spike count per neuron), which is on average 3-7 times lower compared to when *tstep* is 250 ms. This indicates that using a longer time window to encode the input is not necessary to achieve a sufficient level of prediction accuracy, especially when considering implementation on embedded systems.

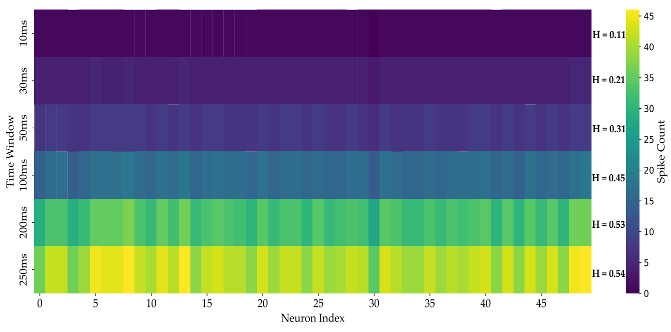

Figure 4. Spike distribution in the reservoir layer for different time step windows. The spike pattern becomes more evenly distributed as the time step window increases

4.4. Effect of Inhibitory Connection on the Network Performance

Another technique to increase variability in spike count patterns is to introduce inhibitory connections between reservoir neurons. In previous simulations, communication between neurons in the reservoir layer was established using only excitatory connections. This means that if a spike occurred on any neuron in the reservoir layer, it consequently led to simultaneous spikes on other neurons. In other words, this reduced the diversity of the spike count patterns. Theoretically, by adding some portion of inhibitory connections in the reservoir layer, the diversity of spike count patterns can be increased. To prove this concept, the connections in the reservoir layer for the network model with *tstep* = [30, 50, 100 ms] were reconfigured with inhibitory connections at ratios of 0.25 and 0.5, selected randomly.

As shown in *figure 5*, for each *tstep* case, adding a portion of inhibitory connections in the reservoir layer generally increased the prediction accuracy. For instance, when the inhibitory ratio was 0.25 (shown by the solid line), the RMSE decreased from 0.097 to 0.093 (for *tstep* = 30 ms) and from 0.092 to 0.089 (for *tstep* = 50 ms). Increasing the inhibitory ratio to 0.5 further reduced the RMSE, as well as the number of synaptic events, which lowered the energy consumption of the network model itself. For example, at *tstep* = 50 ms and a connection density of $p = 0.2$, using a 0.5 inhibitory ratio reduced the network model's energy consumption by approximately 4% (7695 synaptic events, assuming 10 nJ/event). In addition, adding inhibitory connections increased spike distribution.

Table 3. Performance evaluation of RSNN with varying ratios of inhibitory connections

Time Step [ms]	Inhibitory Ratio	RMSE	MAE	Training Loss	Synaptic Event	Spike Distribution
30	0.25	0.093 ± 0.005	0.079 ± 0.004	0.0021	4563	0.28
30	0.5	0.092 ± 0.003	0.078 ± 0.003	0.0018	4463	0.32
50	0.25	0.089 ± 0.004	0.076 ± 0.004	0.0013	7871	0.39
50	0.5	0.087 ± 0.005	0.075 ± 0.005	0.0011	7695	0.41
100	0.25	0.085 ± 0.003	0.075 ± 0.003	0.0007	16164	0.49
100	0.5	0.081 ± 0.004	0.072 ± 0.004	0.0006	15800	0.51

For example, at *tstep* = 100 ms with an inhibitory connection ratio of 0.5, the model achieved a spike distribution exceeding 0.51, which is nearly equivalent to that obtained with a higher *tstep* configuration. The performance evaluation is summarized in *table 3*.

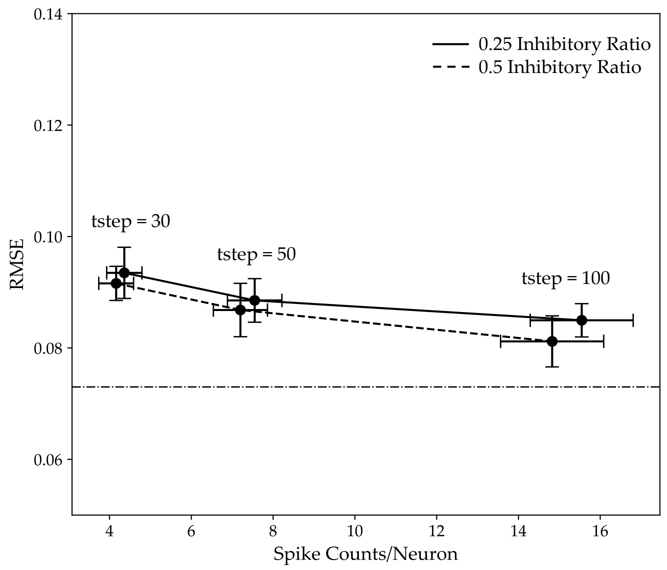

Figure 5. Prediction accuracy (RMSE) versus spike count per neuron. The inclusion of inhibitory connections significantly reduces the RMSE. The solid line represents an inhibitory ratio of 0.25, while the dashed line represents a ratio of 0.5

4.5. Optimize with Readout Layer

Commonly, the RSNN model is designed to work efficiently on low-powered devices, for instance, on embedded SoC-FPGAs or neuromorphic hardware. In this work, the sigmoid transfer function was replaced with the ReLU function, which is more hardware friendly. Furthermore, the performance of the network model was evaluated with different configurations of the readout network. In addition, the default weight initialization is replaced with Kaiming uniform initialization to ensure stable variance and efficient gradient flow across layers, specifically tailored for the ReLU activation function.

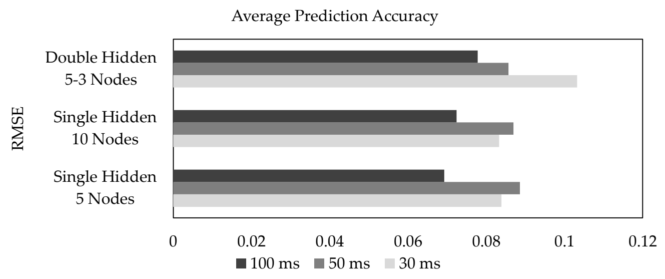

Figure 6. Comparison of prediction accuracy across different readout network configurations. A simple readout network with fewer neurons achieves a low RMSE, comparable to the benchmark value of 0.72, when the network is configured with a higher time step window (*tstep*)

As shown in *figure 6*, reconfiguration of the readout network yields different outcomes. At a lower *tstep* of 30 ms, the performance with a single hidden layer and ReLU activation function provides slightly better accuracy. However, this improvement does not occur when *tstep* is set at 50 ms, where the RMSE worsens compared to the previous setup. Using a higher *tstep* value of 100 ms, accuracy performance improves significantly, outperforming the network at higher *tstep* values (e.g., 250 ms). Nonetheless, utilizing deeper networks with multiple hidden layers did not improve the accuracy performance for any *tstep* value. The overall SoH prediction result (solid line) compared with the calculated value (dashed line) on different battery datasets (B6, B7, and B18) is depicted in *figure 7*. In all cases, the RSNN network model is able to predict the behaviour and the battery's SoH drop pattern efficiently. However, due to variations in battery cells such as internal resistance, capacity, and aging characteristics, some prediction errors are common and expected.

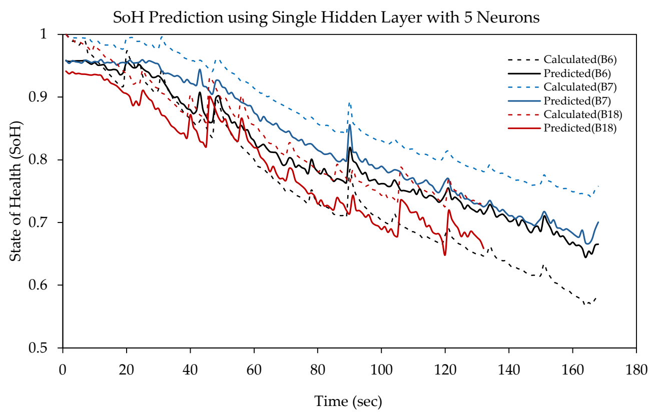

Figure 7. SoH prediction results for different battery datasets using the optimized RSNN. The dashed line represents the calculated values, while the solid line represents the predicted values

Next, a Feedforward Neural Network (FNN) was implemented as a baseline model to benchmark the RSNN's performance. The network consists of three fully connected hidden layers with eight ReLU-activated neurons each, followed by a 25% dropout layer to reduce overfitting. The model was trained using the Adam optimizer with default parameters. The number of training epochs was matched to that of the RSNN for fair comparison. *Table 4* summarizes the results for each battery datasets, evaluated multiple times with different random seeds to account for stochastic variations during training. The RSNN achieved lower prediction errors than the FNN for Battery 6 but showed weaker performance for Batteries 7 and 18. This outcome suggests that Battery 6 exhibits stronger temporal dependencies and nonlinear degradation behavior, which benefit from the RSNN's recurrent dynamics. In contrast, the smoother and more stable degradation patterns of Batteries 7 and 18 are adequately captured by the FNN, where the additional temporal states in the RSNN may introduce unnecessary complexity and slight performance degradation.

Table 4. Prediction accuracy comparison of the FNN and RSNN models for battery SoH estimation

	FNN			RSNN		
Battery	**B6**	**B7**	**B18**	**B6**	**B7**	**B18**
RMSE	0.086 ± 0.004	0.029 ± 0.002	0.019 ± 0.004	0.069 ± 0.011	0.034 ± 0.010	0.029 ± 0.011
MAE	0.074 ± 0.002	0.027 ± 0.003	0.016 ± 0.002	0.062 ± 0.010	0.032 ± 0.010	0.028 ± 0.011

In addition to performance differences, the results also highlight the effect of network complexity. The FNN offers a simpler structure and lower computational demand, which makes it suitable for relatively stable datasets with near-linear degradation patterns. However, this simplicity limits its ability to model temporal dependencies and dynamic variations that often occur in real battery operating conditions. In contrast, the RSNN incorporates recurrent reservoir dynamics that enable temporal processing and adaptive learning from sequential data. This characteristic makes the RSNN inherently more suitable for capturing the nonlinear and time-dependent behaviour of battery degradation, especially under fluctuating load profiles or varying environmental conditions. It is worth noting that the NASA battery dataset, although derived from real experimental measurements, was collected under controlled laboratory conditions with consistent charge–discharge cycles and minimal external noise. As a result, it does not fully represent the variability and uncertainty typically observed in real-world battery applications.

Furthermore, the RSNN offers a biologically inspired framework aligned with emerging neuromorphic computing trends. Its event-driven spiking mechanism enables asynchronous and energy-efficient computation, unlike conventional networks that rely on continuous activation. Although this study simulates the RSNN using the BindsNET library on conventional hardware, its full potential lies in neuromorphic deployment, where sparse spiking activity can substantially reduce power consumption and improve real-time processing. Therefore, while the FNN serves as a useful baseline, the RSNN presents a more scalable and energy-efficient solution for future intelligent battery management systems.

Despite these promising results, the current study is limited to simulation-based evaluation. The performance under real hardware constraints, such as SoC-FPGA or neuromorphic devices, remains unexplored. Factors including memory limitations, fixed-point arithmetic, and spike propagation delays on physical platforms may influence the RSNN's accuracy and energy efficiency. Therefore, future work should focus on implementing and testing the RSNN on real embedded hardware to validate the simulation outcomes and optimize the network for practical deployment in battery management systems.

5. CONCLUSION

In this work, the utilization of RSNN in battery SoH prediction is evaluated in detail. The network model is tested under various configurations to provide a clear understanding of the fundamental structural factors that affect prediction performance. Specifically, this study analyzes the performance impact of varying the number of reservoir neurons, connection density, time step window length, ratio of inhibitory connections, and readout network configuration. The results show that, for battery SoH prediction, a simple RSNN structure is sufficient to achieve high prediction accuracy. The essential aspect is configuring the network to produce diversity in the spike count pattern within the reservoir layer. For the SoH dataset used in this study, the original input features alone are not sufficient to generate high spike pattern diversity. Therefore, careful model configuration is required, guided by observations of spike count diversity in the reservoir layer. Additionally, the findings from this study provide a framework for constructing RSNN models on embedded platforms, whether implemented on neuromorphic devices such as Intel Loihi or on SoC-FPGA platforms with customized hardware circuits. However, further evaluation is required to analyze the network performance under hardware-specific constraints, for instance when reducing the bit length for storing network weight values.

Acknowledgments: The author would like to thank Centre for Research and Innovation Management (CRIM), Universiti Teknikal Malaysia Melaka (UTeM) for its financial support Grant No. PJP/2024/FTKEK/PERINTIS/SA0015 for completing this project.

Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

[1] Hu, X.; Feng, F.; Liu, K.; Zhang, L.; Xie, J.; Liu, B. State estimation for advanced battery management: Key challenges and future trends. Renew. Sustain. Energy Rev. 2019, 114, 109334.

[2] Chen, M.; Ma, G.; Liu, W.; Zeng, N.; Luo, X. An overview of data-driven battery health estimation technology for battery management system. Neurocomputing 2023.

[3] Wang, Y.; Tian, J.; Sun, Z.; Wang, L.; Xu, R.; Li, M.; Chen, Z. A comprehensive review of battery modeling and state estimation approaches for advanced battery management systems. Renew. Sustain. Energy Rev. 2020, 131, 110015.

[4] Ali, M.U.; Zafar, A.; Nengroo, S.H.; Hussain, S.; Alvi, M.J.; Kim, H.-J. Towards a smarter battery management system for electric vehicle applications: A critical review of lithium-ion battery state of charge estimation. Energies 2019, 12, 446.

[5] Ardeshiri, R.R.; Balagopal, B.; Alsabbagh, A.; Ma, C.; Chow, M.-Y. Machine learning approaches in battery management systems: State of the art: Remaining useful life and fault detection. In Proceedings of the 2nd IEEE International Conference on Industrial Electronics for Sustainable Energy Systems (IESES), Cagliari, Italy, 1–3 Sept 2020.

[6] Gu, X.; See, K.; Li, P.; Shan, K.; Wang, Y.; Zhao, L.; Lim, K.C.; Zhang, N. A novel state-of-health estimation for the lithium-ion battery using a convolutional neural network and transformer model. Energy 2023, 262, 125501.

[7] Li, P.; Zhang, Z.; Grosu, R.; Deng, Z.; Hou, J.; Rong, Y.; Wu, R. An end-to-end neural network framework for state-of-health estimation and remaining useful life prediction of electric vehicle lithium batteries. Renew. Sustain. Energy Rev. 2022, 156, 111843.

[8] Ma, G.; Xu, S.; Jiang, B.; Cheng, C.; Yang, X.; Shen, Y.; Yang, T.; Huang, Y.; Ding, H.; Yuan, Y. Real-time personalized health status prediction of lithium-ion batteries using deep transfer learning. Energy Environ. Sci. 2022, 15, 4083–4094.

[9] Fan, Y.; Xiao, F.; Li, C.; Yang, G.; Tang, X. A novel deep learning framework for state of health estimation of lithium-ion battery. J. Energy Storage 2020, 32, 101741.

[10] Zhou, D.; Li, Z.; Zhu, J.; Zhang, H.; Hou, L. State of health monitoring and remaining useful life prediction of lithium-ion batteries based on temporal convolutional network. IEEE Access 2020, 8, 53307–53320.

[11] Li, P.; Zhang, Z.; Xiong, Q.; Ding, B.; Hou, J.; Luo, D.; Rong, Y.; Li, S. State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short-term memory neural network. J. Power Sources 2020, 459, 228069.

[12] Liu, K.; Kang, L.; Xie, D. Online state of health estimation of lithium-ion batteries based on charging process and long short-term memory recurrent neural network. Batteries 2023, 2, 94.

[13] Ma, Y.; Shan, C.; Gao, J.; Chen, H. A novel method for state of health estimation of lithium-ion batteries based on improved LSTM and health indicators extraction. Energy 2022, 251, 123973.

[14] Qu, J.; Liu, F.; Ma, Y.; Fan, J. A neural-network-based method for RUL prediction and SOH monitoring of lithium-ion battery. IEEE Access 2019, 7, 87178–87191.

[15] Reis, D.; Gonçalo, C.; Strange, C.; Yadav, M.; Li, S. Lithium-ion battery data and where to find it. Energy AI 2021, 5, 100081.

[16] Wang, H.; Li, Y.-F.; Zhang, Y. Bioinspired spiking spatiotemporal attention framework for lithium-ion batteries state-of-health estimation. Renew. Sustain. Energy Rev. 2023, 188, 113728.

[17] Wang, H.; Sun, M.; Li, Y.-F. A brain-inspired spiking network framework based on multi-time-step self-attention for lithium-ion batteries capacity prediction. IEEE Trans. Consum. Electron. 2023.

[18] Pfeiffer, M.; Pfeil, T. Deep learning with spiking neurons: Opportunities and challenges. Front. Neurosci. 2018, 12, 774.

[19] Nunes, J.D.; Carvalho, M.; Carneiro, D.; Cardoso, J.S. Spiking neural networks: A survey. IEEE Access 2022, 10, 60738–60764.

[20] Yao, M.; Zhao, G.; Zhang, H.; Hu, Y.; Deng, L.; Tian, Y.; Xu, B.; Li, G. Attention spiking neural networks. IEEE Trans. Pattern Anal. Mach. Intell. 2023.

[21] Zhang, R.; Leng, L.; Che, K.; Zhang, H.; Cheng, J.; Guo, Q.; Liao, J.; Cheng, R. Accurate and efficient event-based semantic segmentation using adaptive spiking encoder–decoder network. IEEE Trans. Neural Netw. Learn. Syst. 2024.

[22] Yamazaki, K.; Vo-Ho, V.-K.; Bulsara, D.; Le, N. Spiking neural networks and their applications: A review. Brain Sci. 2022, 12, 863.

[23] Dey, S.; Banerjee, D.; George, A.M.; Mukherjee, A.; Pal, A. Efficient time series classification using spiking reservoir. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 18–23 July 2022.

[24] Bing, Z.; Meschede, C.; Röhrbein, F.; Huang, K.; Knoll, A.C. A survey of robotics control based on learning-inspired spiking neural networks. Front. Neurorobot. 2018, 12, 35.

[25] Prognostics Center of Excellence—Data Repository. NASA Ames Prognostics Research Center. Available online: https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository (accessed on 27 August 2025).

[26] Merolla, P.A.; Arthur, J.V.; Alvarez-Icaza, R.; Cassidy, A.S.; Sawada, J.; Akopyan, F.; Modha, D.S. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science* 2014, 345, 668–673.

[27] Frenkel, C.; Lefebvre, M.; Legat, J.D.; Bol, D. A 0.086-mm² 12.7-pJ/SOP 64k-synapse 256-neuron online-learning digital spiking neuromorphic processor in 28-nm CMOS. *IEEE Trans. Biomed. Circuits Syst.* 2018, 13, 145–158.

[28] Thakur, C.S.; Molin, J.L.; Cauwenberghs, G.; Indiveri, G.; Kumar, K.; Qiao, N.; Etienne-Cummings, R. Large-scale neuromorphic spiking array processors: A quest to mimic the brain. *Front. Neurosci.* 2018, 12, 891.