Indonesian Physical Review
Volume 09 Issue 03, September 2026
P-ISSN: 2615-1278, E-ISSN: 2614-7904

A Hybrid TCN-LSTM Model for Predictive Maintenance of AWS Power Supplies

Marzuki Sinambela[1*], Rifqi Daffa Ul-haq[2], Dibyo Susanto[1], Agustina Rachmawardani[1]
[1] *Program in Applied Instrumentation Meteorology, Climatology, and Geophysics, STMKG, Indonesia.*
[2] *Undergraduate Program in Applied Instrumentation Meteorology, Climatology, and Geophysics, STMKG, Indonesia.*

Corresponding Author's E-mail: sinambela.m@gmail.com

Article Info

Article info:
Received: 25-07-2025
Revised: 29-06-2026
Accepted: 06-07-2026

Keywords:
Forecast; Power Supply; TCN; LSTM

How To Cite:
M. Sinambela, R. D. Ul-haq, D. Susanto, A. Rachmawardani, "A Hybrid TCN-LSTM Model for Predictive Maintenance of AWS Power Supplies", Indonesian Physical Review, vol. 9, no. 3, p 469-488, 2026.

DOI:
https://doi.org/10.29303/ipr.v9i3.544.

Abstract

Reliable power supply units are essential for Automatic Weather Stations (AWS) to maintain continuous data collection. However, traditional maintenance schedules often fail to prevent sudden equipment downtime. While machine learning can enable predictive maintenance, standard standalone models typically struggle to capture both immediate short-term anomalies and slow, long-term degradation. To address this gap, this study aims to evaluate and propose a hybrid Temporal Convolutional Network (TCN) and Long Short-Term Memory (LSTM) architecture specifically designed for AWS power supply forecasting. Using empirical time-series data, we monitored five operational parameters at 10-minute intervals from September 2023 to November 2024. Correlation analysis established battery temperature as a primary health indicator due to its strong inverse relationship with voltage ($r = -0.87$). Comparative evaluations demonstrated that while individual TCN and LSTM models exhibited architectural trade-offs, the proposed hybrid TCN-LSTM model achieved the highest predictive accuracy ($R^2 = 0.9497$; MAPE = 0.05%). The findings confirm that integrating these networks effectively balances rapid anomaly detection with stable long-term trend forecasting. Practically, this hybrid model can be integrated into AWS telemetry systems as a robust diagnostic tool, providing automated early warnings to prevent critical power failures.

Introduction

Automatic Weather Stations (AWS) are integrated systems that gather and store real-time weather data, making subsequent processing much easier [1]. Because their readings support aviation, ports, agriculture, and disaster mitigation, AWS networks are strategically important. Indonesia's meteorological agency (BMKG) operates more than 100 stations every day, and any failure, especially in the power supply unit, can bring a station offline. As the sole energy source for all other AWS components, the power supply demands close, proactive attention from technicians. An effective maintenance schedule, therefore, needs to spot early warning signs before minor issues escalate into serious faults. Modern equipment has grown so complex that guessing when something will break is no longer practical. Data-driven

predictive maintenance, built on historical records and careful analysis, lets engineers monitor component health in real time [2]. Done well, it extends service life, cuts operating costs, and boosts overall efficiency [3]. Achieving these benefits hinges on technologies such as machine learning that can detect patterns in past data and support accurate decision-making.

Traditional maintenance strategies, such as preventive and corrective maintenance, often lead to unplanned downtime, higher repair costs, and reduced equipment life [4]. Machine learning (ML)-based predictive maintenance is therefore essential to ensuring AWS reliability, especially in remote areas where power supply failures can disrupt continuous data acquisition. In an AWS, operational variables such as battery voltage, battery temperature, and load current form rich time-series data. Processing these streams with ML techniques enables early detection of sensor anomalies and helps avert component failures before they occur [5], [6]. Among existing models, Temporal Convolutional Network (TCN) and Long Short-Term Memory (LSTM) excel at time series modeling: the TCN layer captures subtle short-term fluctuations, while the LSTM layer preserves long-term memory and cyclical patterns [7], [8], [9]. These capabilities are relevant to AWS power supply maintenance, where both long-range usage trends and short-term events, such as voltage sag, extreme temperature rise, or sudden power surges, can signal impending system failure [10]. Their combination, therefore, enables the model to generalize over both short-term and long-term forecasting without sacrificing predictive accuracy [9], [11]. In addition, the hybrid TCN-LSTM architecture is straightforward to implement and can be deployed in a wide range of industrial contexts, including system modeling, time series forecasting, and predictive maintenance [12].

Recent research underscores the potential of machine learning to strengthen predictive maintenance strategies across diverse industrial settings. Gopali et al. [13] compared TCN and LSTM models on the Secure Water Treatment (SWaT) dataset and reported that TCN delivered higher anomaly detection accuracy and shorter training times, although both networks remained competitively close. Complementing this, Paliling and Sudirman. [14] demonstrated a Random Forest pipeline from data collection and preprocessing to feature selection, model evaluation, and deployment for machine condition monitoring. Their model exceeded 90% accuracy, enabling earlier interventions, lower operating costs, and longer equipment life. In the meteorological domain, Santoso et al. [15] developed an LSTM-based system to identify sensor faults in an AWS using five years of 10-minute interval data from Jatiwangi, Indonesia; the model achieved high accuracy (above 90% for most sensors) with a low false alarm rate, highlighting the value of historical data for timely maintenance actions.

However, a distinct methodological and applicative gap remains unaddressed in these prior studies. While existing predictive maintenance research successfully applies standalone deep learning models, they often face a fundamental architectural trade-off when handling the complex time-series data typical of AWS power supplies. Standalone convolutional models (such as TCNs) excel at capturing abrupt, short-term anomalies, such as sudden voltage drops caused by extreme weather conditions or instantaneous load spikes, but tend to introduce oscillatory noise. Conversely, recurrent networks (such as LSTMs) are highly effective at smoothing data and preserving long-term cyclical trends, such as gradual battery degradation, but react too slowly to sharp spikes. Previous studies have rarely explored how these deep learning architectures can be specifically adapted to monitor the unique, dual-stressor operational environment of meteorological power units.

To address this gap, this study contributes beyond a standard model comparison by proposing a specialized hybrid TCN-LSTM predictive framework tailored for AWS power supply maintenance. By leveraging the dilated causal convolutions of the TCN as an early warning stage for sharp anomalies, and feeding its output into an LSTM layer to stabilize seasonal fluctuations, this architecture is designed to capture both the immediate and long-term behavioral changes of the power unit. Ultimately, the objective is not merely to evaluate the performance trade-offs of these networks, but to establish that the synergistic TCN-LSTM configuration provides a robust, mathematically balanced diagnostic tool. This specialized approach ensures a highly reliable forecasting mechanism that can be directly implemented into BMKG's operational telemetry to prevent unexpected station downtime.

Theory and Calculation

Maintenance Systems

Maintenance practices have traditionally relied on reactive responses to existing breakdowns or preventive inspections conducted on strictly established schedules. However, these conventional maintenance methods often result in unpredictable equipment failures, time-consuming procedures, and unnecessary repair interventions. To optimize operations, modern maintenance strategies aim to extend the operational lifespan of essential assets, enhance workplace safety, and minimize unexpected production stoppages. By transitioning to data-driven predictive maintenance, organizations can successfully execute repair operations only when future equipment failures are genuinely anticipated, thereby reducing overall operational expenses and avoiding pointless downtime [16].

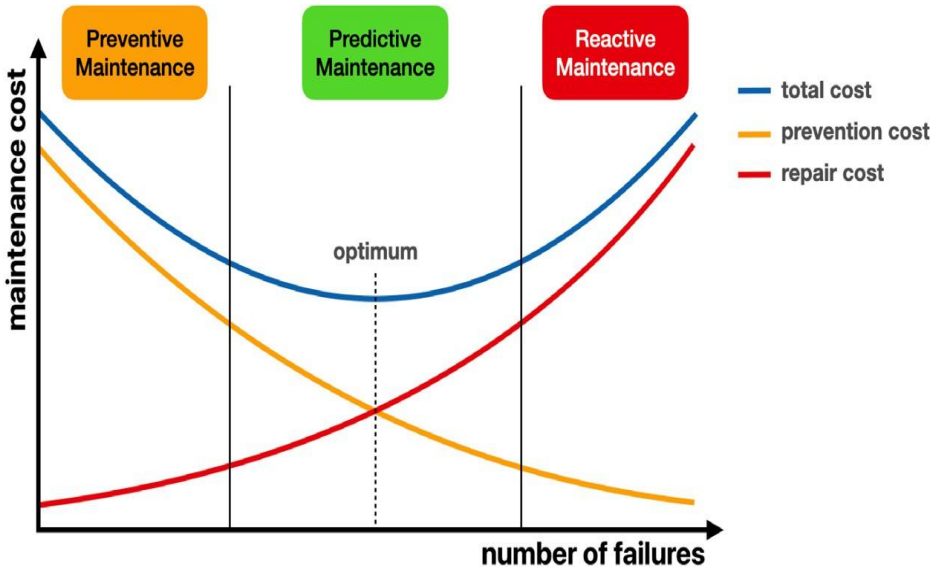


Figure 1. The difference between preventive maintenance, predictive maintenance, and corrective maintenance systems [17].

Preventive maintenance relies on predefined schedules such as fixed time intervals or operational cycles to trigger routine servicing and maintenance procedures. While this approach aims to avert functional breakdowns, it often fails to detect underlying and unmeasured defects within the system. Consequently, relying solely on scheduled maintenance can result in premature, expensive component replacements that still have significant remaining useful life [18]. Predictive maintenance, in contrast, uses forecasting

tools to determine the right moment for service. It relies on continuous condition monitoring, so interventions occur only when necessary. This approach enables early detection of potential faults by leveraging historical data analysis with machine learning, inferential statistics, and related predictive techniques [19]. Corrective maintenance is carried out after the equipment can no longer operate normally. It involves diagnosing the root cause, resetting machine controls, and taking other steps to restore optimal function. Unlike the other strategies, corrective maintenance is applied only when a device has already failed, making it unsuitable for assets that are still running [20].

Temporal Convolutional Network

Temporal Convolutional Networks (TCNs) are fully convolutional, one-dimensional CNN variants designed for sequence modeling. In many time-series tasks, they have outperformed recurrent approaches such as RNNs and LSTMs because they handle long receptive fields while preserving temporal order. A TCN follows two core rules: the output length must match the input length, and information cannot leak from the future to the past. To meet these requirements, a TCN stacks fully convolutional layers that combine causal and dilated convolutions with residual blocks, yielding high parallelism, stable gradients, and a flexible receptive field, advantages not typically available in LSTM architectures [21].

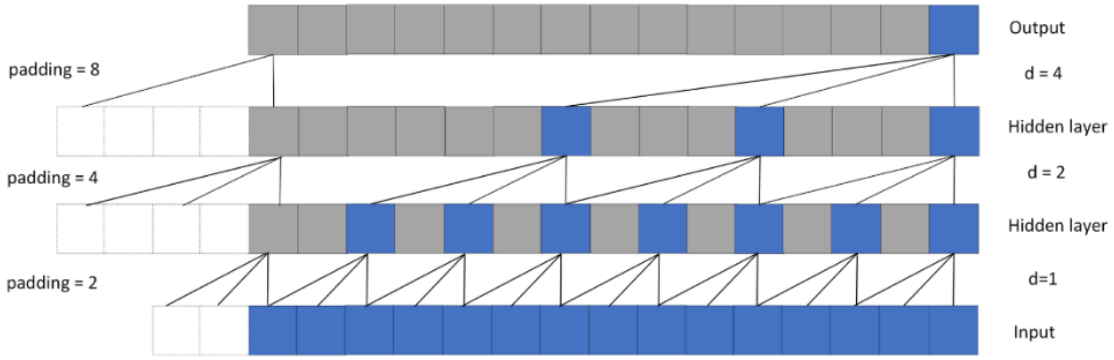


Figure 2. Dilated causal convolution structure with several dilation rates of 1,2,4 and a kernel size of 3 [21].

TCNs employ dilated causal convolutions to enlarge the receptive field by skipping input values at fixed intervals. This allows each filter to "see" a wider context than its kernel size alone would permit, making the network more effective at capturing temporal patterns. Residual blocks modify only part of the input, preserving the original feature information. In a TCN, the outputs of dilated causal convolutions are added back to the block input to produce the final block output [22].

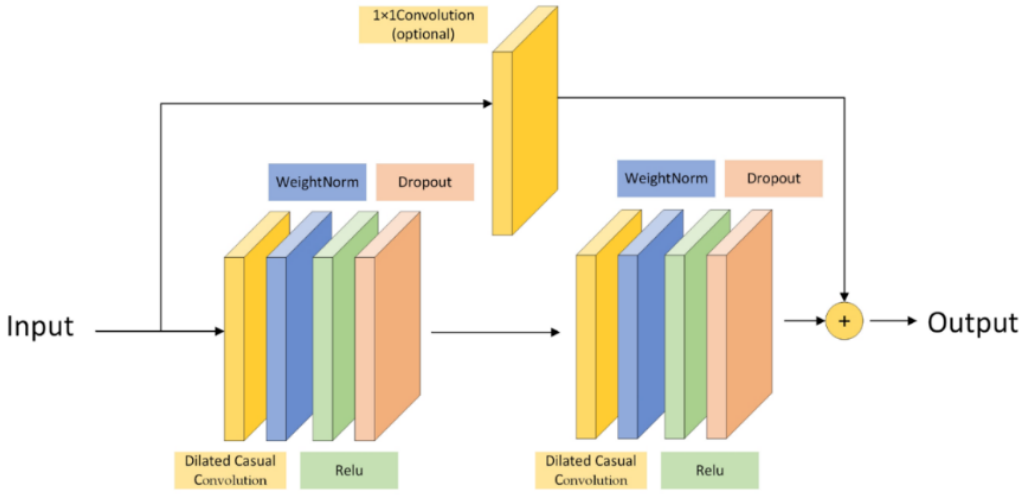


Figure 3. Residual block structure [21].

As the number of layers increases, the volume of extracted features grows, but deep networks often struggle with vanishing gradients. Residual blocks help mitigate this problem while further extending the receptive field [23].

Long Short-Term Memory

Long Short-Term Memory (LSTM) networks were developed to overcome the limitations of standard recurrent neural networks in handling long-term dependencies within sequential data. By introducing specialized gate functions into the cell structure, LSTMs effectively capture information across large input gaps, leading to their widespread adoption and the emergence of numerous network variants [24]. LSTM is a specialized form of Recurrent Neural Network (RNN) that can retain long-term information through memory cells that update the previous hidden state and respond to incoming data at each neuron. An RNN may learn temporal correlations across long sequences because its output is dependent on both the current input and weights, as well as earlier inputs. With internal memory cells and gating mechanisms, LSTMs address the exploding and vanishing gradient problems that often hinder training in conventional RNNs. The four primary components of an LSTM cell are the cell state, input gate, forget gate, and output gate. The three gates control how information is stored and updated in the cell state [25].

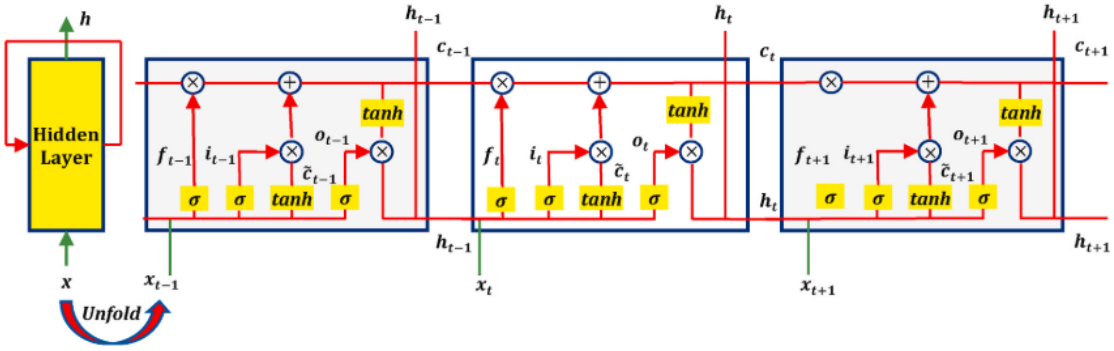


Figure 4. Internal structure of LSTM [25].

As illustrated in Figure 4, the LSTM cell contains neuron layers (circles) and element-wise operations (squares). Black arrows illustrate the flow of information within the cell, between cells, and at the cell output. Each cell produces two outputs: the hidden state (h_t), which is passed to the next cell and also serves as part of that cell's input, and the cell state (C_t), which is the core memory element. The cell state is depicted as a horizontal line running through the LSTM, connecting all outputs, as shown in Figure 4. The forget gate decides which information to discard from the cell, the input gate determines what new information to write, and the output gate decides what part of the internal state to expose as the cell's output [26].

Hyperparameters

Most machine learning algorithms have hyperparameters: model settings specified manually before training that control how the algorithm learns. Their values cannot be derived directly from the data; they must be defined explicitly when the model is set up. In other words, hyperparameters need to be fixed before the learning process begins, and they strongly influence the model's output [27]. Artificial neural networks include several tunable hyperparameters, such as the learning rate, the number of units per layer, the number of layers, the number of training epochs, the chosen loss function, the activation functions, and others.

Evaluation Metrics

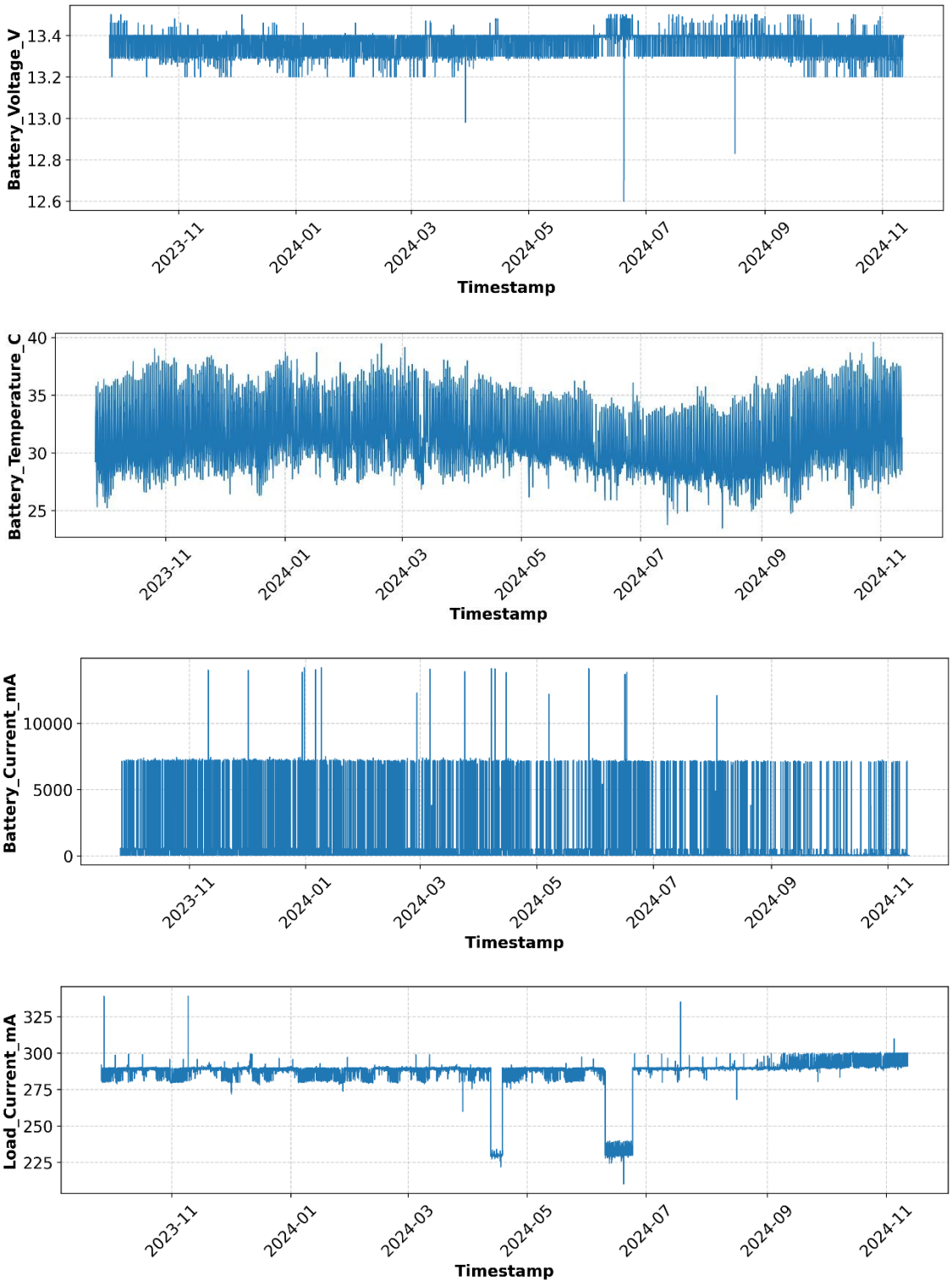
The predictive capability of the models was evaluated through five widely adopted statistical metrics: Coefficient of Determination (R^2 Score, Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) [28]. The detailed mathematical formulations and variable definitions for each of these metrics are comprehensively provided in the Supplementary Information.

Experimental Method

Dataset Description

This study uses historical power supply data from the digital Automatic Weather Station (AWS) in Denpasar, sampled every ten minutes from September 2023 to November 2024. The input variables comprise battery voltage, battery current, battery temperature, load current, and charging voltage.

Figure 5 shows that battery voltage remained highly stable at about 13.2 to 13.4 V from October 2023 to October 2024. Even so, several sharp drops below 13 V (down to roughly 12.6 V) are evident around March, June, and August 2024. Battery temperature readings fluctuate more widely, ranging from 25°C to nearly 40°C, over the same period. A distinct seasonal trend emerges: temperatures peak in late 2023 and early 2024, decline toward mid-2024, and rise again toward the end of the study window. Battery current measurements average roughly 7,000 mA but display occasional extremes: brief dips to nearly zero and spikes above 14,000 mA. Load current is comparatively steady at 280-300 mA between October 2023 and October 2024, although periodic drops below 240 mA occur, especially from April to July 2024. The charging voltage remained consistently near 16 V, with no notable excursions, indicating that the charging subsystem operated within normal limits throughout the observation period.


Battery_Voltage_V
13.4
13.2
13.0
12.8
12.6
2023-11
2024-01
2024-03
2024-05
2024-07
2024-09
2024-11
Timestamp
Battery_Temperature_C
40
35
30
25
2023-11
2024-01
2024-03
2024-05
2024-07
2024-09
2024-11
Timestamp
Battery_Current_mA
10000
5000
0
2023-11
2024-01
2024-03
2024-05
2024-07
2024-09
2024-11
Timestamp
Load_Current_mA
325
300
275
250
225
2023-11
2024-01
2024-03
2024-05
2024-07
2024-09
2024-11
Timestamp

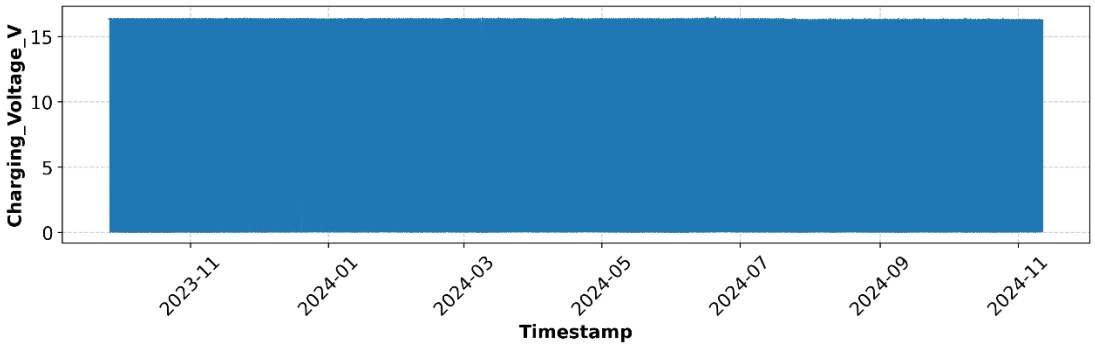


Figure 5. AWS power supply parameters.

Table 1. Dataset descriptive statistics.

Parameter	Count	Mean	Std	Min	50%	Max
Battery Voltage (V)	59,473	13.36	0.05	12.6	13.39	13.5
Battery Temperature (°C)	59,473	31.28	2.91	23.47	30.53	39.58
Battery Current (mA)	59,473	364	1,081.45	12	49.3	14,196.8
Load Current (mA)	59,473	286.12	12.94	210.1	289.4	339.1
Charging Voltage (V)	59,473	8.47	8.02	0.01	15.08	16.5

Table 1 presents descriptive statistics for the power supply parameters. Overall, the battery management system appears relatively stable on several variables, although some show high variability. Battery voltage averages 13.36 V with a tiny standard deviation (0.05 V), indicating tight control, with most readings between 13.30 and 13.40 V. The minimum and maximum values (12.60 and 13.50 V) still fall within acceptable operating limits. Battery temperature is also fairly steady, averaging 31.28°C with a 2.91°C standard deviation, even though it fluctuates between 23.47°C and 39.58°C. These swings remain reasonable when environmental factors and system load are taken into account. A very different pattern appears in battery current. The mean is 364.00 mA, but the standard deviation is extremely high at 1081.45 mA, reflecting large swings from as low as 12.00 mA to peaks of 14,196.80 mA. In contrast, load current is much more stable, averaging 286.12 mA with a standard deviation of only 12.94 mA; the 210.10–339.10 mA range suggests limited fluctuation in power consumption. Charging voltage shows a bimodal-like distribution. While the maximum reaches 16.50 V and the upper quartile indicates many values above 15 V, the mean is only 8.47 V with a substantial standard deviation (8.02 V). The near-zero minimum (0.01 V) implies two operating states: active charging (high voltage) and idle or no charging (very low or zero voltage). This pattern supports the assumption that the charging system operates periodically and responds to the battery's condition.

Data Preprocessing and Feature Selection

Before the data were fed into the model, a preprocessing stage was performed to enhance data quality and improve predictive performance. An initial inspection confirmed that the dataset contained no missing or null values; therefore, data imputation was not required. A

rigorous feature selection process was subsequently conducted to prevent the introduction of oscillatory noise and reduce computational overhead. Based on the preliminary correlation analysis, battery voltage and battery temperature showed a strong inverse relationship ($r = -0.87$) and were thus selected as the primary input features. Conversely, variables such as load current and battery current demonstrated negligible correlations ($r < 0.1$) with the target variable and were deliberately excluded from the input array. The selected features (battery voltage and temperature) were scaled to a 0-1 range using a MinMaxScaler to ensure gradient stability. To define the model's input-output structure, the scaled variables were structured into sequences using a sliding window approach with a lookback period of 30 timesteps. This temporal sequence allows the network to learn the multivariate historical fluctuations over the preceding 30 intervals. For the output, the target label (battery voltage) was mapped directly in its original, unscaled format at a 1-timestep forecast horizon. This ensures the model learns from scaled multivariate inputs while outputting exact, real-time voltage predictions. Finally, the structured dataset was partitioned using chronological splitting into a combined training and validation subset (70%) and a completely unseen test subset (30%).

Table 2. Dataset partitioning results.

Dataset	Amount of Data
Training	33,304
Validation	8,327
Testing	17,842
Total	59,473

Model Implementation and Training Environment

All analytical and deep learning procedures in this study were implemented in Python utilizing a Jupyter Notebook environment. The experiments were executed on a local workstation equipped with an Intel Core i5 processor, 16 GB RAM, and an NVIDIA RTX 3050 GPU. The neural network architectures were constructed using the TensorFlow and Keras frameworks, while the TCN layers were explicitly integrated using the external keras-tcn library. To ensure an unbiased performance comparison, all experimental configurations were trained using a fixed batch size of 32. The validation strategy relied on chronological splitting, where the validation subset was used solely to monitor generalization and trigger early stopping (with a patience of 5 to 10 epochs), thereby preventing data leakage from the test set. Instead of random trial-and-error, the hyperparameter search procedure involved systematic empirical tuning over predefined ranges, specifically testing learning rates between [0.001 and 0.0001] and dropout rates between [0.05 and 0.10], optimized using the Adam optimizer to minimize the Mean Squared Error (MSE) loss function.

Implementation of Initial TCN and LSTM Models

The baseline TCN and LSTM models were built with deliberately simple, matched architectures to enable a fair performance comparison. The LSTM comprises two stacked layers: 32 units in the first layer and 64 in the second, followed by a Dense (1) output layer. The TCN mirrors this depth, employing two convolutional blocks with 32 and 64 filters and dilation rates of [1, 2]. Both models were compiled with the Adam optimizer (default learning

rate = 0.001) and mean squared error (MSE) as the loss metric. This balanced design lets us evaluate predictive accuracy and computational efficiency on equal terms. Training was limited to 40 epochs and controlled by early stopping, with validation loss as the main criterion and a patience of 5 epochs; training halts if no improvement is observed across 5 consecutive epochs. In both baselines, battery voltage and temperature served as input features, with battery voltage as the target.

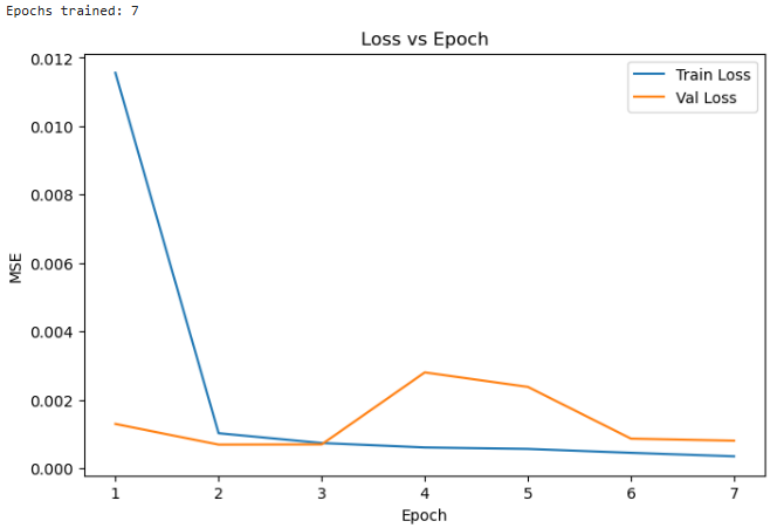


Figure 6. TCN initial model training period.

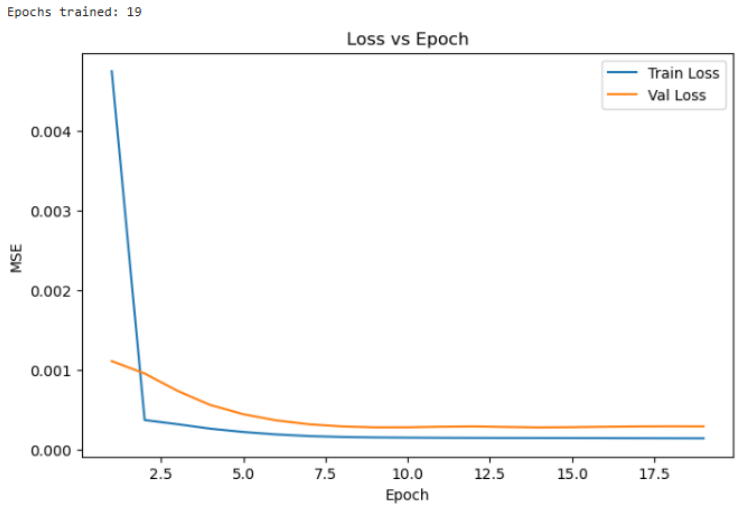


Figure 7. LSTM initial model training period.

Implementation Final TCN and LSTM Models

The final TCN and LSTM models were refined with deeper, more expressive architectures. For each network, we increased capacity to 128 filters in the TCN and 128 units in the LSTM, added an extra hidden layer of 64 units, and introduced a dropout (0.05 on the first two layers, 0.10 on the third) to curb overfitting. The TCN now uses four dilation rates [1, 2, 4, 8] while retaining the original kernel size and single-stack layout; both models apply ReLU activation

in hidden layers and a linear activation at the output. Compilation employed the Adam optimizer with a smaller learning rate (0.0001) and mean squared error as the loss metric. Dropout values and the learning rate adjustment were selected empirically through trial and error. Training ran for up to 50 epochs with an early stopping patience of 10, terminating if the validation loss failed to improve for 10 consecutive epochs.

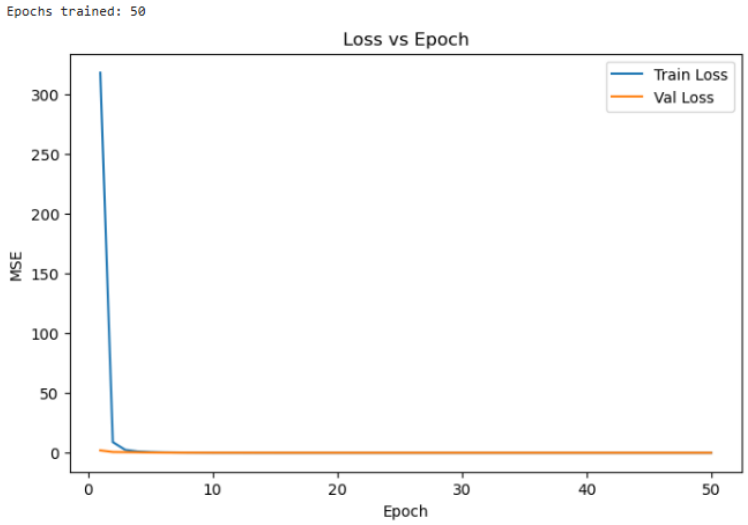


Figure 8. TCN final model training period.

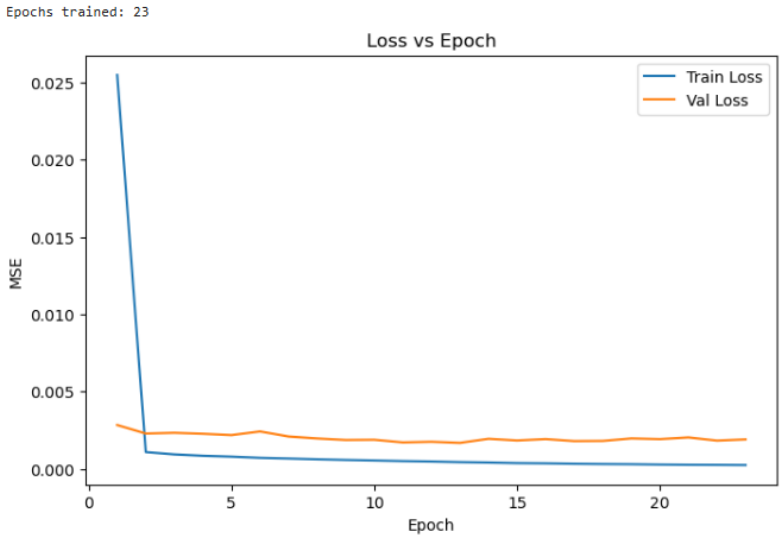


Figure 9. LSTM final model training period.

Implementation of Hybrid TCN-LSTM Models

The hybrid TCN-LSTM model starts with two dilated TCN blocks (32 and 64 filters; dilation rates of 1, 2, 4, and 8), each with 0.05 dropout regularization. These blocks serve as an early warning stage, capturing sharp anomalies in the input sequence, such as voltage spikes or abrupt temperature drops. Their output feeds a 128-unit LSTM layer with 0.10 dropout, which smooths seasonal fluctuations and stabilizes the signal. A hidden layer of 64 units with

ReLU activation then provides the final non-linear transformation before the linear output node. This cascade is designed to respond quickly to anomalies while maintaining a smooth, consistent prediction curve.

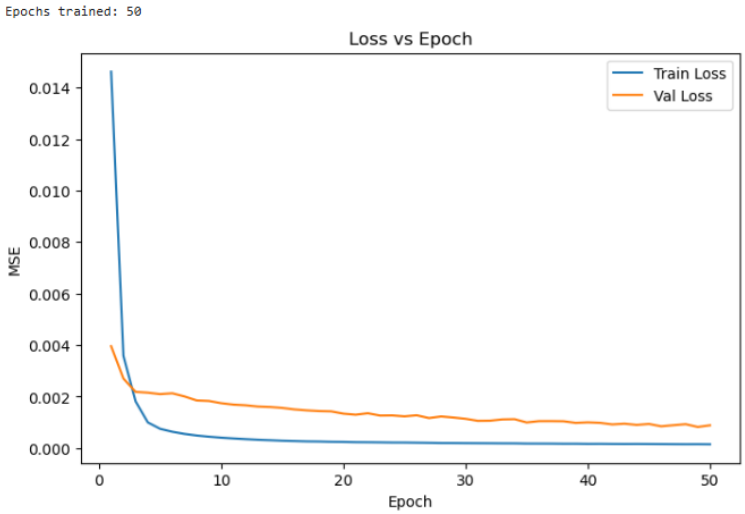


Figure 10. Hybrid TCN-LSTM model training period.

Result and Discussion

Correlation Matrix

This study visualizes the correlation matrix as a heatmap, making it easy to see which parameter pairs influence one another and to what extent. Coefficients close to +1 or -1 signal a strong relationship, whereas values near zero indicate little or no association.

Figure 11 shows that correlations among the power supply variables range from -0.87 to +0.67. The strongest link is a negative one between battery voltage and battery temperature (r = -0.87): when the battery warms, its voltage tends to drop sharply, and vice versa. Battery temperature also shows a fairly strong positive correlation with charging voltage (r = 0.67), so higher temperatures are associated with higher charger output. Load current, by contrast, is essentially independent; its largest coefficients, 0.13 with battery temperature and 0.058 with charging voltage, are negligible. Including load current as an input variable would therefore inject noise and could even degrade model performance. Battery current is similarly uncorrelated, peaking at only 0.068 with battery voltage and showing small negative values elsewhere. Finally, battery voltage is moderately and inversely related to charging voltage (r = -0.59), indicating that an increase in charger voltage is often accompanied by a decrease in stored battery voltage.

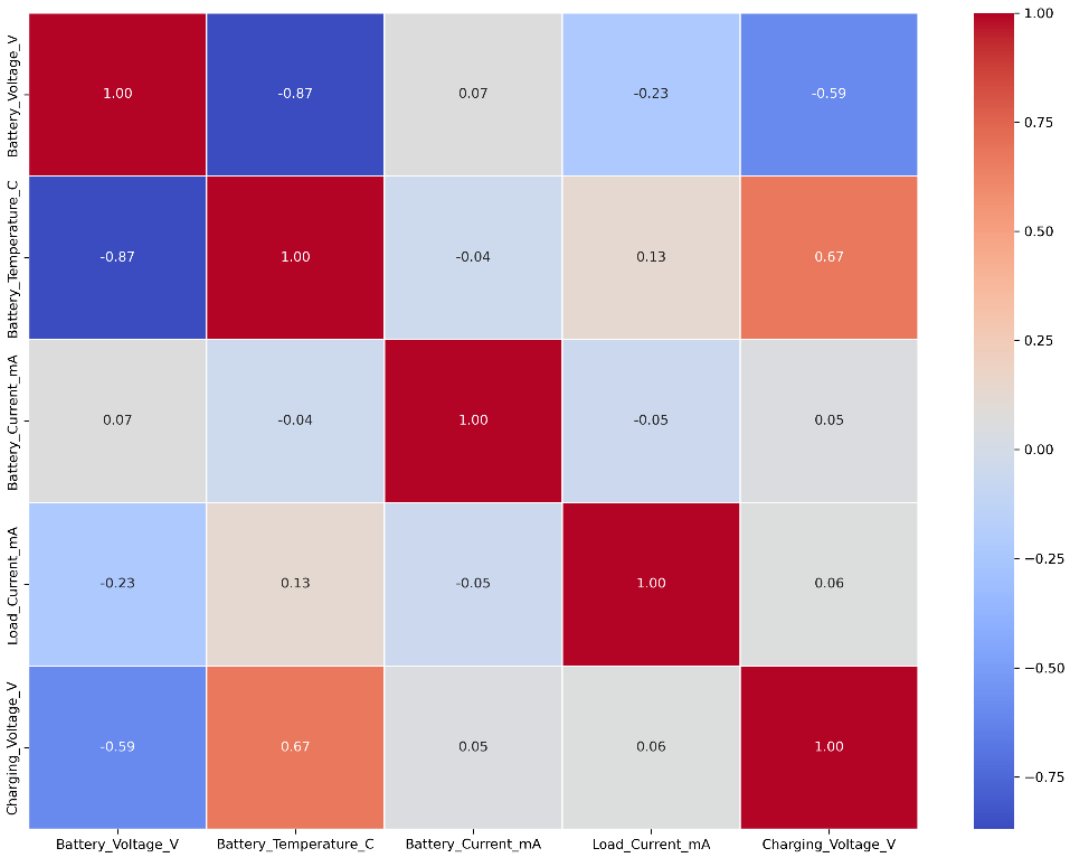


Figure 11. Heatmap correlation.

Model Prediction

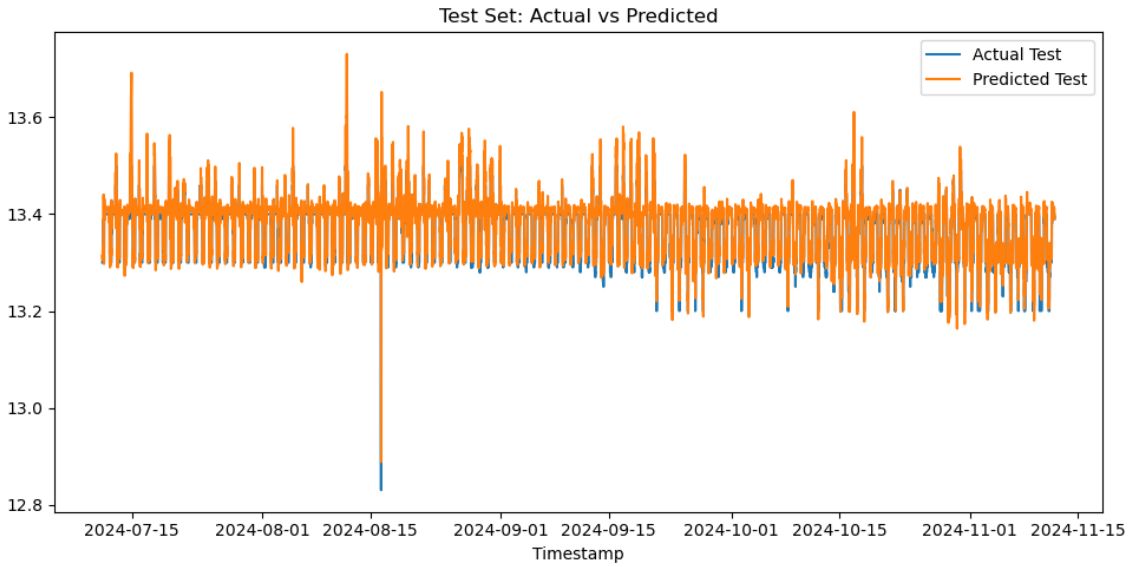


Figure 12. Predicted vs actual TCN initial model.

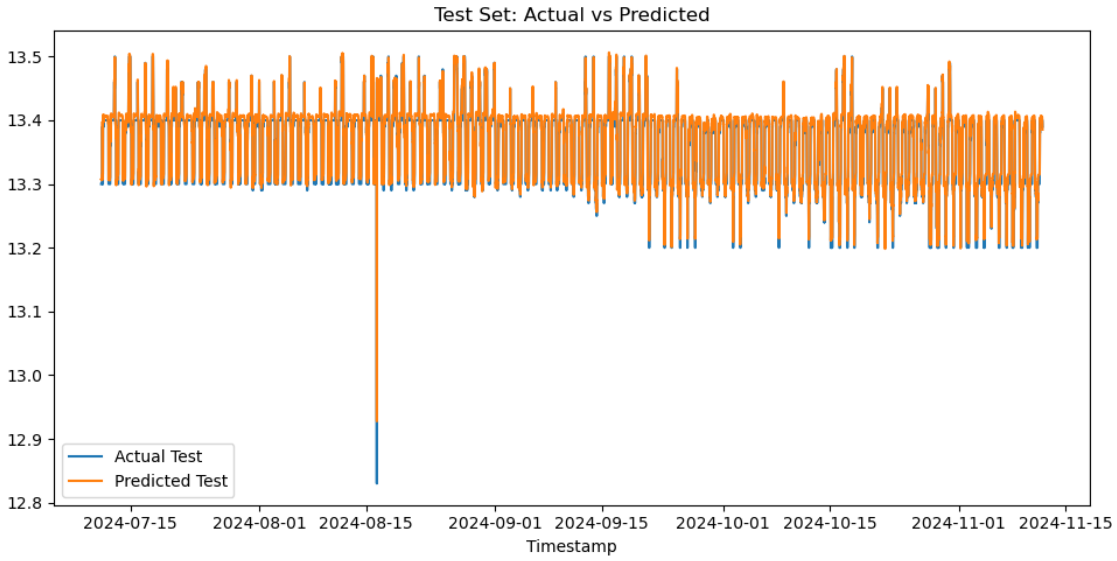


Figure 13. Predicted vs actual LSTM initial model.

Figures 12 and 13 compare the predicted and actual battery voltage curves produced by the baseline TCN and LSTM models. In Figure 12, the TCN responds almost instantaneously to voltage spikes. When a sharp drop occurs in mid-August, its forecast mirrors the decline and even briefly overshoots it before settling back into the normal range. At voltage peaks, the TCN likewise tends to edge slightly above the measured values, indicating that its two dilated blocks (32 and 64 filters, dilations [1 2]) excel at capturing abrupt changes, albeit with some extra oscillation beyond the ground truth. Figure 13 shows a notably smoother, more stable LSTM output. During extreme anomalies, the predicted descent and recovery are gentler, rarely overshooting the actual readings. The network's two LSTM layers (32 and 64 units) effectively dampen sudden swings, yielding a less "sharp" but more consistent curve with minimum noise. Taken together, these results show that the baseline TCN is better suited to detecting anomalies, while the baseline LSTM provides steadier forecasts with fewer spurious fluctuations.

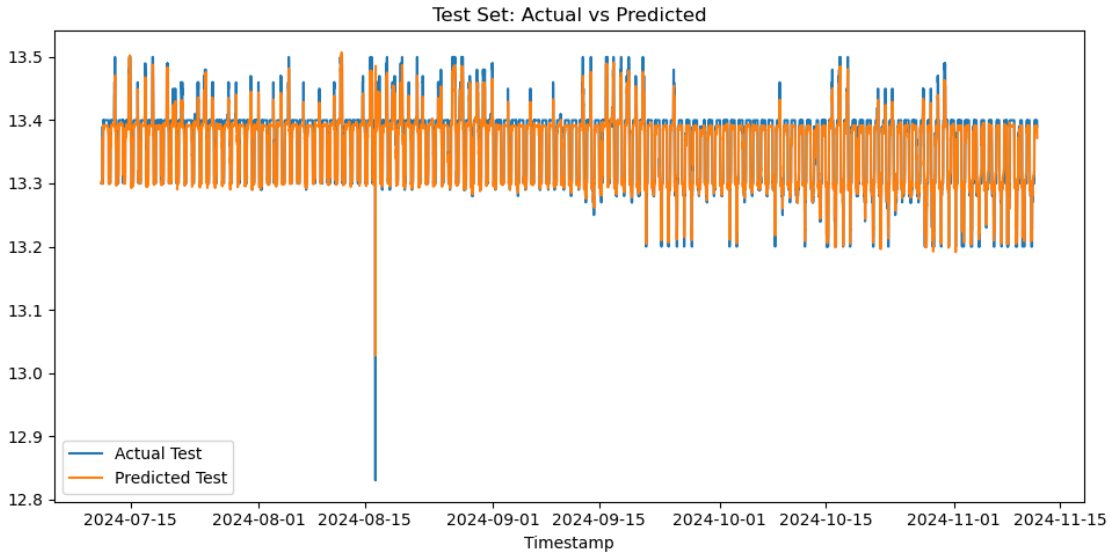


Figure 14. Predicted vs actual TCN final model.

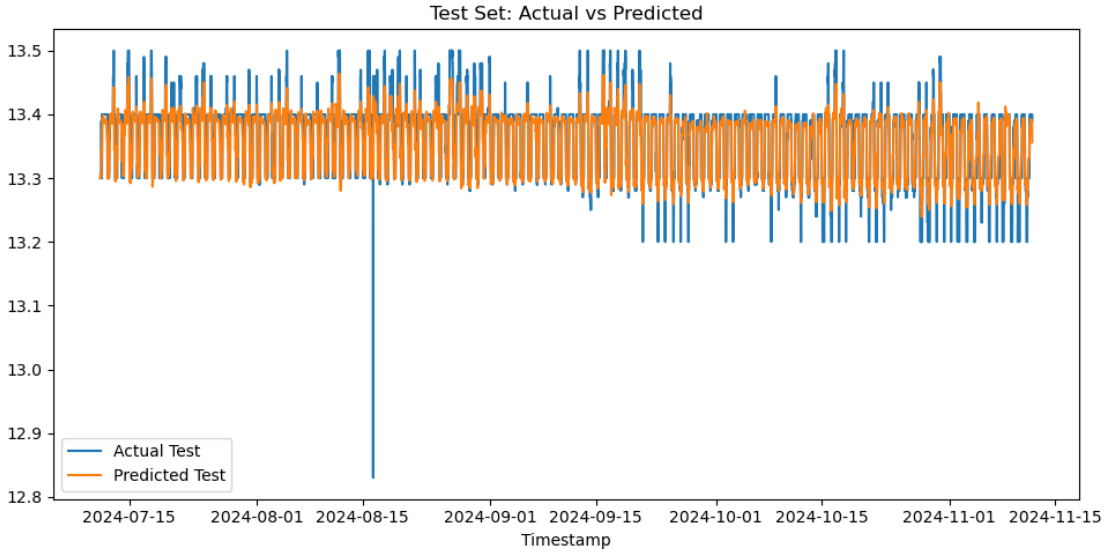


Figure 15. Predicted vs actual LSTM final model.

Figures 14 and 15 compare the predicted and measured battery voltage traces produced by the final TCN and LSTM configurations. In Figure 14, the TCN now employs three convolutional blocks with dilations [1, 2, 4, 8] and light dropout (0.05 in the first two layers, 0.10 in the third). The network remains highly responsive to abrupt events. For example, it tracks the sharp mid-August drop almost instantly, although the predicted trough is slightly shallower than before. At the same time, recurring seasonal swings are reproduced faithfully without the excessive overshoot seen in the baseline, showing that a longer receptive field can suppress noise while preserving sensitivity to sudden change. Figure 15 shows the effect of enlarging the LSTM to 128 units, adding dropout, reducing the learning rate, and inserting a 64-unit hidden layer with ReLU activation. The enhanced model reacts a bit more slowly to extreme anomalies than

the baseline, and its seasonal oscillations are damped, yielding a smoother curve. The trade-off is a loss of fine detail at the peaks and valleys. In short, the baseline LSTM better captures both seasonal patterns and sharp deviations, whereas the final LSTM offers a steadier, low-noise forecast at the cost of pinpoint accuracy during brief spikes.

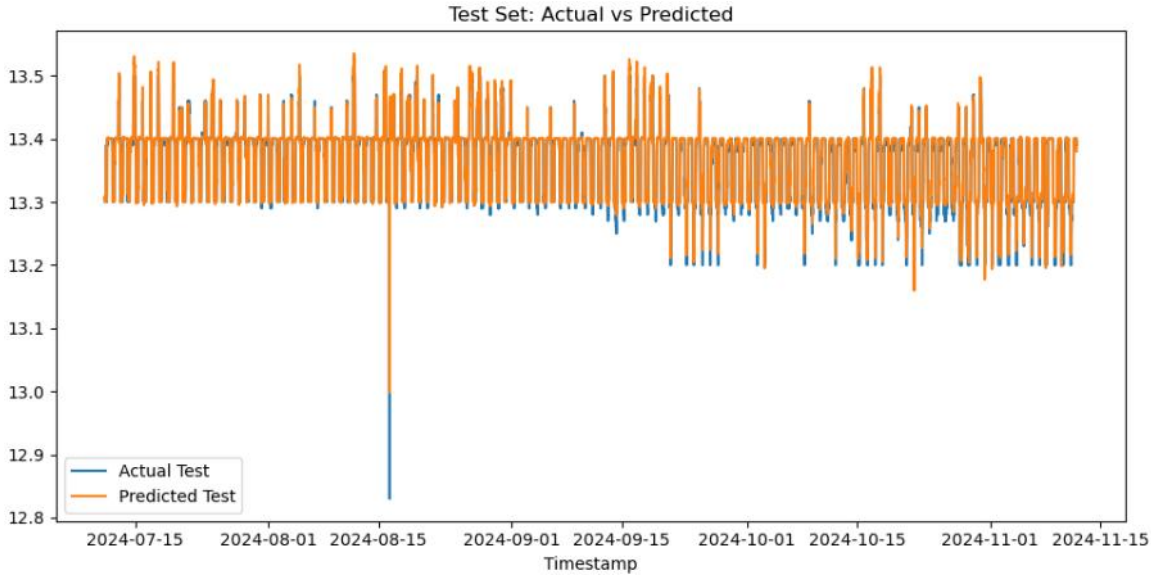


Figure 16. Predicted vs actual hybrid TCN-LSTM model.

In Figure 16, the predicted curve closely overlays the actual data, confirming that the model captures seasonal patterns and short-term fluctuations with high fidelity. It also handles anomalies consistently, introducing little additional noise. Overall, the hybrid TCN–LSTM combines the LSTM's ability to smooth and preserve long-range trends with the TCN's rapid response to abrupt changes. The result is a prediction line that stays accurate at the peaks and troughs while remaining stable across recurring seasonal cycles.

Model Evaluation

Table 3. Evaluation of initial and final models.

Method	Initial Model		Final Model	
	TCN	LSTM	TCN	LSTM
R^2 Score	0.8038	0.9463	0.9416	0.8475
MSE	0.0007	0.0002	0.0002	0.0005
RMSE	0.0257	0.0135	0.0141	0.0227
MAE	0.0184	0.0087	0.0093	0.0152
MAPE	0.0014	0.0007	0.0007	0.0011

Table 3 shows that the baseline LSTM delivers the strongest overall performance among the initial single models, posting the highest R^2 score (0.9463) and the lowest error metrics. By contrast, the baseline TCN underperforms, reaching only 0.8038 in R^2. However, after hyperparameter tuning, the final TCN improves markedly (R^2 rises to 0.9416), while the final LSTM's performance noticeably degrades (R^2 slips to 0.8475, and MSE climbs to 0.0005). This divergence in tuning response highlights a fundamental architectural characteristic. TCN benefits significantly from increased network capacity, more filters, and higher dilation rates, as the expanded receptive field enables the model to capture longer historical contexts without losing sensitivity to sudden local spikes. In contrast, increasing the depth and complexity of an LSTM network for a relatively stable time-series dataset often leads to overfitting and over-smoothing. The added layers likely caused the LSTM to aggressively smooth out the subtle, high-frequency local patterns inherent to the AWS battery data, ultimately degrading its predictive accuracy.

Table 4. Evaluation of the hybrid TCN-LSTM model.

R^2 Score	MSE	RMSE	MAE	MAPE
0.9497	0.0002	0.0130	0.0061	0.0005

Table 4 indicates that the hybrid TCN-LSTM architecture achieves a clear performance jump over the preceding four configurations. Recording an R^2 score of 0.9497, this integrated network explains nearly 95% of the variance in the operational data. However, relying on absolute metric values alone can be statistically misleading. It is crucial to contextualize the extremely low MAPE values (0.05%–0.14%) observed across all models. While mathematically accurate, these microscopic percentages are primarily a byproduct of the dataset's nature. Because battery voltage operates within a highly restricted and stable range (averaging 13.36 V), the denominator in the MAPE formula is consistently massive relative to the small absolute errors (~0.006 V). Consequently, the resulting percentage is naturally minimized. Therefore, while MAPE confirms the absence of catastrophic deviations, metrics that evaluate variance explanation, such as the R^2 score, are far more indicative of the model's true forecasting capabilities.

To confirm the statistical significance of the error reduction, we conducted a confidence analysis alongside a Diebold Mariano (DM) test, comparing the proposed hybrid model against the best-performing baseline (the initial LSTM).

Table 5. Statistical significance and confidence analysis.

Model	95% Confidence Interval (MAE)	DM Statistic	p-value	Significance
Initial LSTM	[0.0066, 0.0069]	-	-	-
Hybrid TCN-LSTM	[0.0056, 0.0060]	16.2048	< 0.001	Significant

As summarized in Table 5, computing the 95% Confidence Intervals (CI) for the Mean Absolute Error (MAE) revealed that the hybrid model maintained a tighter and strictly lower error bound of [0.0056, 0.0060]. In contrast, the initial LSTM yielded a higher error range of

[0.0066, 0.0069]. Because these two confidence intervals do not overlap at all, this provides strong evidence that the hybrid configuration systematically outperforms the baseline. This conclusion is further corroborated by the DM test, which yielded a test statistic of 16.2048 and a p-value of < 0.001. This deeper interpretation reveals that the hybrid architecture successfully overcomes the limitations of single models discussed in previous literature. While earlier works, such as Gopali et al. [13], concluded that standalone TCN and LSTM are merely competitive alternatives, and Santoso et al. [15] relied solely on standalone LSTM for AWS sensor diagnostics, our findings demonstrate that AWS power supplies require a synergistic approach. By utilizing the TCN as a rapid response filter for sharp anomalies (e.g., instantaneous voltage drops due to erratic load or weather) and the LSTM to stabilize underlying seasonal degradation trends, this study establishes that hybrid modeling is not just an incremental upgrade, but a theoretical necessity for highly stable yet anomaly-prone operational environments.

Conclusion

This study demonstrates that battery voltage and battery temperature are strongly coupled, exhibiting an inverse correlation of -0.87. Temperature also serves as a primary health indicator for the AWS power supply, given its significant links with other operational variables. Among single-model baselines, the standard LSTM achieved high predictive accuracy ($R^2 = 0.9463$, MAPE = 0.07%), outperforming both the untuned TCN and the hyperparameter-tuned LSTM. Ultimately, integrating these approaches into a hybrid TCN-LSTM framework maximized performance ($R^2 = 0.9497$, MAPE = 0.05%). The findings confirm that this hybrid architecture is highly superior at balancing rapid anomaly detection with stable long-term forecasting.

Translating these findings into operational practice involves integrating the hybrid model into the existing AWS telemetry infrastructure. The model can operate continuously as a backend diagnostic tool, triggering automated early warning alerts on a centralized maintenance dashboard whenever the forecasted voltage deviates from safe thresholds. Despite these promising practical implications, this study has several limitations that must be acknowledged. The model was trained on single-station data from Denpasar and relies on limited variables, which may not fully capture the diverse failure modes observed in other regions. Furthermore, there is no external validation using independent datasets, and the evaluation was conducted solely in an offline environment with no real deployment testing. Therefore, recommendations for future work include conducting multi-station validation across diverse climate zones, developing anomaly classification to diagnose specific hardware faults, and executing real-time predictive maintenance implementation directly on edge computing devices at the station level.

Acknowledgment

The authors gratefully acknowledge the invaluable contributions that made this study possible. Special thanks go to the Directorate of Instrumentation and Calibration, Indonesian Agency for Meteorology, Climatology, and Geophysics (BMKG), for providing the AWS power-supply data that served as the cornerstone of this work. We also extend our sincere appreciation to the Meteorology, Climatology, and Geophysics College (STMKG) for its

ongoing academic guidance, institutional support, and essential research facilities, all of which were critical to the successful completion of this study.

References

[1] D. Hikmah, L. E. Arisanti, and D. Irmawan, "Tipe Pasang Surut di Pelabuhan Benoa Bali dengan Metode Admiralty Berdasarkan Data Automatic Weather Station (AWS)," *J. Widya Climago*, vol. 2, pp. 86–95, 2020, [Online]. Available: https://ejournal-pusdiklat.bmkg.go.id/index.php/climago/article/view/28

[2] N. H. A. Wahab *et al.*, "Systematic review of predictive maintenance and digital twin technologies challenges, opportunities, and best practices," *PeerJ Comput. Sci.*, vol. 10, 2024, doi: 10.7717/PEERJ-CS.1943.

[3] H. H. Hosamo, P. R. Svennevig, K. Svidt, D. Han, and H. K. Nielsen, "A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics," *Energy Build.*, vol. 261, p. 111988, 2022, doi: 10.1016/j.enbuild.2022.111988.

[4] S. P. S. Rathore, A. Gupta, A. Parashar, D. Upadhyay, R. K. Deb, and A. Gupta, "Machine Learning in Predictive Maintenance for Industrial Equipment," *Proc. - IEEE 2024 1st Int. Conf. Adv. Comput. Commun. Networking, ICAC2N 2024*, pp. 1537–1541, 2024, doi: 10.1109/ICAC2N63387.2024.10895271.

[5] L. Lin, C. Walker, and V. Agarwal, "Explainable machine-learning tools for predictive maintenance of circulating water systems in nuclear power plants," *Nucl. Eng. Technol.*, vol. 57, no. 9, p. 103588, 2025, doi: 10.1016/j.net.2025.103588.

[6] H. Li, W. Zhao, Y. Zhang, and E. Zio, "Remaining useful life prediction using multi-scale deep convolutional neural network," *Appl. Soft Comput. J.*, vol. 89, p. 106113, 2020, doi: 10.1016/j.asoc.2020.106113.

[7] L. Ren, Y. Liu, X. Wang, J. Lu, and M. J. Deen, "Cloud-Edge-Based Lightweight Temporal Convolutional Networks for Remaining Useful Life Prediction in IIoT," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12578–12587, 2021, doi: 10.1109/JIOT.2020.3008170.

[8] C. Y. Hsu, Y. W. Lu, and J. H. Yan, "Temporal Convolution-Based Long-Short Term Memory Network With Attention Mechanism for Remaining Useful Life Prediction," *IEEE Trans. Semicond. Manuf.*, vol. 35, no. 2, pp. 220–228, 2022, doi: 10.1109/TSM.2022.3164578.

[9] A. H. Almaliki and A. Khattak, "Short- and long-term tidal level forecasting: A novel hybrid TCN + LSTM framework," *J. Sea Res.*, vol. 204, no. January, p. 102577, 2025, doi: 10.1016/j.seares.2025.102577.

[10] A. Boujamza and S. Lissane Elhaq, "Attention-based LSTM for Remaining Useful Life Estimation of Aircraft Engines," *IFAC-PapersOnLine*, vol. 55, no. 12, pp. 450–455, 2022, doi: 10.1016/j.ifacol.2022.07.353.

[11] J. Bi, X. Zhang, H. Yuan, J. Zhang, and M. C. Zhou, "A Hybrid Prediction Method for Realistic Network Traffic With Temporal Convolutional Network and LSTM," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1869–1879, 2022, doi: 10.1109/TASE.2021.3077537.

[12] X. Wang, Y. Liu, X. Liang, C. Zhang, C. Yang, and W. Gui, "Learning an Enhanced TCN-LSTM Network for Temperature Process Modeling in Rotary Kilns," *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 3056–3067, 2025, doi: 10.1109/TASE.2024.3388709.

[13] S. Gopali, F. Abri, S. Siami-Namini, and A. S. Namin, "A Comparative Study of Detecting Anomalies in Time Series Data Using LSTM and TCN Models," pp. 1–15,

2021, [Online]. Available: http://arxiv.org/abs/2112.09293
[14] F. Paliling and Z. Sudirman, "Machine Learning untuk Perawatan Prediktive Mesin Berbasis Random Forest," *INFINITY*, vol. 3, no. 2, pp. 80–84, 2023, doi: 10.34148/infinity.v9i1.xxx.
[15] B. Santoso *et al.*, "Predictive Maintenance Automatic Weather Station Sensor Error Detection using Long Short-Term Memory," *Ultim. Comput. J. Sist. Komput.*, vol. 15, no. 2, pp. 41–51, 2023, doi: 10.31937/sk.v15i2.3403.
[16] S. Zeb and S. K. Lodhi, "AI FOR PREDICTIVE MAINTENANCE: REDUCING DOWNTIME AND ENHANCING EFFICIENCY," *Enrich. J. Multidiscip. Res. Dev.*, vol. 3, no. 1, pp. 135–150, 2025, doi: 10.55324/enrichment.v3i1.338.
[17] D. Pagano, "A predictive maintenance model using Long Short-Term Memory Neural Networks and Bayesian inference," *Decis. Anal. J.*, vol. 6, no. January, p. 100174, 2023, doi: 10.1016/j.dajour.2023.100174.
[18] H. Taoufyq, K. El Guemmat, K. Mansouri, and F. Akef, "Predictive Maintenance Approaches : A Systematic Literature Review," *J. Ind. Eng. Manag.*, vol. 18, no. 3, pp. 427–458, 2025, doi: https://doi.org/10.3926/jiem.8537.
[19] S. Lourensius, N. H. Djanggu, and Y. E. Prawatya, "Implementasi Predictive Maintenance untuk Mesin Pengupas Buah Pinang dengan Mikrokontroler," *Integr. Ind. Eng. Manag. Syst.*, vol. 7, no. 2, pp. 1–6, 2023, [Online]. Available: https://jurnal.untan.ac.id/index.php/jtinUNTAN/issue/view/2162
[20] C. T. N. Siregar, P. Kindangen, and I. D. Palandeng, "Evaluasi Pemeliharaan Mesin dan Peralatan Produksi PT. Multi Nabati Sulawesi (MNS), Kota Bitung, *J. EMBA , J. Ris. Ekon. Manajemen, Bisnis dan Akunt.*, vol. 10, no. 3, p. 428, 2022, doi: 10.35794/emba.v10i3.42362.
[21] J. Yao, Z. Cai, Z. Qian, and B. Yang, "A novel approach based on TCN-LSTM network for predicting waterlogging depth with waterlogging monitoring station," *PLoS One*, vol. 18, no. 10 October, pp. 1–19, 2023, doi: 10.1371/journal.pone.0286821.
[22] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," 2018, [Online]. Available: http://arxiv.org/abs/1803.01271
[23] Y. Zuo *et al.*, "Short text classification based on bidirectional TCN and attention mechanism," *J. Phys. Conf. Ser.*, vol. 1693, no. 1, 2020, doi: 10.1088/1742-6596/1693/1/012067.
[24] Y. Yu, X. Si, C. Hu, and J. Zhang, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures," *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, 2019, doi: https://doi.org/10.1162/neco_a_01199.
[25] N. Somu, G. Raman M R, and K. Ramamritham, "A deep learning framework for building energy consumption forecast," *Renew. Sustain. Energy Rev.*, vol. 137, no. April 2020, p. 110591, 2021, doi: 10.1016/j.rser.2020.110591.
[26] K. Smagulova and A. P. James, "A survey on LSTM memristive neural network architectures and applications," *Eur. Phys. J. Spec. Top*, vol. 228, no. 10, pp. 2313–2324, 2019, doi: 10.1140/epjst/e2019-900046-x.
[27] W. Nugraha and A. Sasongko, "Hyperparameter Tuning on Classification Algorithm with Grid Search," *Sistemasi*, vol. 11, no. 2, p. 391, 2022, doi: 10.32520/stmsi.v11i2.1750.
[28] K. Karthika, P. Balasubramanie, K. Dharshini, P. Shanmugapriya, and T. E. Ramya, "`," *2024 15th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2024*, pp. 1–8, 2024, doi: 10.1109/ICCCNT61001.2024.10723986.