Vol. 08, No. 3 (2026) 1831-1846, doi: 10.24874/PES08.03.033

Proceedings on Engineering Sciences

www.pesjournal.net

LEVERAGING HYBRID MODEL FOR RESOURCE ALLOCATION AND PREDICTION IN DYNAMIC EDGE ENVIRONMENT

Shreya Agarwal [1]
Sonam Seth

Received 05.05.2026.
Revised 17.06.2026.
Accepted 23.07.2026.

Keywords:

Hybrid BiLSTM-GRU networks, Attention-enhanced deep learning, Edge computing, Resource allocation and prediction, Time-series forecasting

A B S T R A C T

Edge computing has emerged from the broader paradigm of distributed processing to bring computation closer to end devices, but the dynamic and resource-constrained nature of edge workloads makes resource allocation, utilization forecasting, and energy-aware scheduling considerably more difficult than in centralized cloud settings. Conventional statistical and single-branch recurrent forecasting methods may not jointly capture short-term fluctuations, long-range temporal dependencies, and context-sensitive temporal importance. To address this limitation, the present study proposes an attention-enhanced residual hybrid deep-learning framework, referred to as Proposed-Mega, which integrates a one-dimensional convolutional front end, bidirectional LSTM and GRU processing, multi-head self-attention, layer normalization, dropout regularization, and a persistence-based residual connection for resource/energy forecasting. Extensive preprocessing, temporal feature engineering, and stationarity testing using the Augmented Dickey-Fuller test were performed. On the held-out test set, Proposed-Mega achieves the lowest MAE (0.1137), MSE (0.0198), and RMSE (0.1406), and the highest R^2 (0.6855) among the five evaluated architectures. The Hybrid model achieves the lowest MAPE (9.9566%), while Proposed-Mega records 10.0245%. When the five metrics are considered jointly using average rank, Proposed-Mega obtains the best overall ranking (1.20), followed by Hybrid (1.80). Five-fold cross-validation of Proposed-Mega gives RMSE values from 0.1679 to 0.2073 kW, with a mean of 0.1804 ± 0.0138 kW. These findings indicate that the attention-enhanced hybrid architecture provides strong predictive performance on the held-out dataset and offers a useful predictive basis for proactive resource-management decisions, while full closed-loop resource allocation remains outside the scope of the present evaluation.

© 2026 Published by Faculty of Engineering

[1] Corresponding author: Shreya Agarwal
Email: shreya.gupta1606@gmail.com

1. INTRODUCTION

Edge computing fundamentally changed the traditional paradigm of computing from centralized computation to decentralization by pulling data processing closer to where end users and Internet-of-Things (IoT) devices connect. Although traditional cloud-centric architectures provided a vast processing power and storage capacity, they are still restricted by the high latencies that result from bandwidth-utilization overheads, and service response times which can only improve up to practical limits for applications requiring real time or near-real-time response (Shi et al. 2016) Edge computing solves these limitations through local processing from the data origin or nearby, which reduces redundancy in moving data over a network towards centralized servers, increases system responsiveness and enhances operational performance (Sartzetakis et al., 2024). Resource utilization has been one of the most significant challenges faced by a heterogeneous environment due to limited computational capacity at individual edge nodes, the variability of workloads they are designed to process and their diverse connected edge devices (Satyanarayanan, 2017; Yu et al., 2018). As depicted in Figure 1, which shows a typical edge-computing architecture along with its primary application domains, we have a three-layered relationship among the end devices, edge nodes and centralized cloud infrastructure that introduces the resource-allocation problem considered in this work.

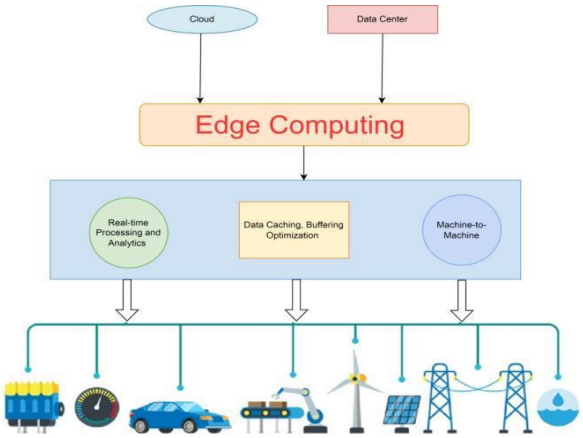


Figure 1. Edge Computing Architecture and Applications

Static thresholding, rule-based heuristics or reactive scaling policies such as triggering execution of a workload only after detection of resource constraints violation have been the common methods used in conventional resource-management solutions for dynamic edge environments. These kinds of approaches suffer from the disadvantages of low prediction accuracy and bad adaptability or scalability under time-varying workloads (Mach & Becvar, 2017; Mao et al., 2017), since they do not predict future demand and are therefore unable to re-allocate computational memory or bandwidth resources proactively before congestion or energy inefficiency occurs. It has a particularly serious impact on the edge as you might lack compute headroom and both under-provisioning service degradation) and over-provisioning waste of energy are costly.

To address these limitations, machine learning and deep-learning methods have been progressively adopted for intelligent resource management. Historical utilization data allow machine-learning models to uncover latent temporal patterns and forecast future resource demand, while hybrid methods that combine learning algorithms with optimization techniques further improve prediction precision and adaptivity (Li et al., 2021). Hybrid prediction frameworks have, in particular, been shown to be effective for resource allocation in edge and cloud environments: Sangani et al. (2024) proposed an error-optimization and resource-prediction algorithm to reduce energy consumption in cloud environments, and Gongada et al. (2024) integrated a convolutional neural network with a fruit-fly optimization technique for intelligent resource allocation. These studies confirm the growing prominence of hybrid and optimization-based models in contemporary computing environments, and they motivate the exploration of deep hybrid recurrent architectures for the specific problem of edge resource forecasting. Deep Neural Networks (DNNs), loosely inspired by the layered structure of biological neural systems, are widely adopted in this space for their capacity to model complex, nonlinear, and temporal relationships in large-scale data through hierarchical feature learning trained via backpropagation and gradient-descent optimization (Abbas et al., 2018). In edge-computing context, DNNs can be leveraged to predict workloads and resource demand which allow for proactive management but are generally accompanied pairs with lightweight or compact variants due to their high computational complexity which makes them impossible deploy on resource-constrained edge nodes (Yuan et al., 2019).

However, pure single-branch deep-learning models suffer from some innate limitations in the context of dynamic edge workloads. In standard recurrent architectures such as unidirectional LSTM networks, information is processed in only the forward direction during sequence processing which constrains any influence of future context when interpreting a given time step; purely convolutional models learn local patterns well but are poorly suited to modeling long-range temporal dependencies; and models without an explicit attention mechanism consider all time steps equally by assigning uniform weight across the entire sequence reducing capacity to focus on relevant temporal regions: be it spikes in demand or daily/weekly seasonal peaks that are most important for producing accurate forecasts. Hybrid prediction models that integrate machine learning, deep learning and multi-objective optimization techniques have the potential to yield better flexibility, scalability and robustness for such dynamic settings (Deng et al., 2016), but the literature review shows that extant hybrid recurrent approaches for edge and cloud resource

forecasting have never hitherto been jointly combined bidirectional recurrent branches, a multi-head self-attention mechanism and residual-learning into one formally specified architecture evaluated on fine-grained energy-consumption data.

This condition is the motivation for one of the main design choices in this paper a hybrid architecture, connecting a BiLSTM branch to a BiGRU branch. To retain the BiLSTM branch to support explicit cell-state memory, which is advantageous for capturing longer-range dependencies and mitigating vanishing-gradient effects over very long sequences, while we keep the BiGRU branch because of its relatively low parameter count and fast convergence that mitigates some of the additional computational cost from layering a convolutional front-end before our scope layer as well as an attention layer. Combining the two, rather than relying on either alone, is intended to balance representational capacity against training efficiency in a manner appropriate for edge-oriented forecasting workloads, and the multi-head self-attention layer is added specifically to let the network learn a data-driven weighting over time steps rather than treating all positions in the input window as equally informative.

Taken together, the reviewed literature indicates that existing studies have primarily focused on either statistical/ML-based prediction, single-branch deep-learning forecasting, or optimization-driven resource allocation in isolation, while comparatively limited attention has been given to formally specified, attention-enhanced hybrid recurrent architectures that unify short-term fluctuation modeling, long-range dependency capture, and residual robustness within one forecasting framework for dynamic edge environments. This constitutes the research gap addressed in this paper, and it is elaborated further, together with its logical chain from existing limitations to the proposed solution.
To close this gap, the present study proposes an attention-enhanced residual hybrid BiLSTM-BiGRU forecasting framework called Proposed-Mega for intelligent resource prediction in dynamic edge-computing environments. The framework combines a one-dimensional convolutional feature-extraction stage, bidirectional recurrent processing, a multi-head self-attention layer, a persistence-based residual connection, layer normalization, and dropout regularization, culminating in a dense output layer for resource/energy-consumption forecasting. The framework is evaluated against LSTM, BiLSTM, BiGRU, and a non-attention Hybrid baseline on the Individual Household Electric Power Consumption dataset, using MAE, MSE, RMSE, R^2, and MAPE as evaluation metrics on training and held-out test partitions, together with five-fold cross-validation of the proposed model's RMSE.

The key contributions of this research are as follows:

- A formally specified, attention-enhanced hybrid recurrent architecture named Proposed-Mega that integrates convolutional feature extraction, recurrent temporal modelling, multi-head self-attention, and persistence-based residual learning within a single forecasting framework for dynamic edge environments.
- An empirical evaluation of Proposed-Mega against four recurrent baselinesLSTM, Bi LSTM, BiGRU, and a non-attention Hybrid model using MAE, MSE, RMSE, R^2, and MAPE on both training and held-out test partitions, together with five-fold cross-validation of the proposed model's RMSE.
- A controlled model-development comparison across the five evaluated architectures to characterize their training and development-set behaviour, together with an explicit statistical-validation framework for future repeated-run evaluation.
- A discussion connecting forecasting-accuracy improvements to their practical implications for proactive, prediction-driven resource allocation in edge-computing deployments.

To distinguish architectural novelty from implementation and application contributions, the study emphasizes the joint integration of established recurrent, convolutional, attention, and residual components and evaluates their combined behaviour on the selected forecasting problem. The resource allocation and prediction in edge and fog computing, organized thematically around classical resource-management strategies, machine-learning-based prediction, deep-learning-based prediction, LSTM-based approaches, and GRU-based and hybrid architectures, concluding with a synthesis that motivates the research gap.

2. LITERATURE REVIEW

For the past decade, a large amount of work has been directed to resource utilization prediction and resource allocation in edge cloud and fog computing environments including classical distributed-systems approaches, machine-learning-based prediction methods, but more recently also deep-learning and hybrid optimization frameworks. The relevant literature by theme, describes the method, contribution and limitations of each study and concludes with a synthesis leading into the research gaps articulated.

2.1 Resource Allocation in Edge and Fog Computing

Shi et al. (2016) provided a foundational survey establishing the vision and open challenges of edge computing, arguing that pushing computation toward the network edge reduces latency and bandwidth consumption relative to centralized cloud processing, but that doing so introduces new challenges in resource-constrained, heterogeneous deployment settings. Satyanarayanan (2017) traced the emergence of edge

computing as a response to the limits of centralized cloud architectures for latency-sensitive applications, while Yu et al. (2018) surveyed edge-computing techniques specifically for the Internet of Things, cataloguing architectural variants and open research problems related to resource management under device heterogeneity. Mach and Becvar (2017) and Mao et al. (2017) each surveyed mobile-edge-computing architectures from complementary perspectives computation offloading and communication, respectively and both identified resource allocation under dynamic, latency-sensitive workloads as a persistent open problem. Dab et al., (2019) addressed this problem directly by jointly optimizing task offloading and resource allocation for mobile edge computing through an integer-programming formulation that minimizes energy consumption subject to system-efficiency constraints their approach demonstrates the value of joint optimization but, as an optimization-based rather than learning-based method, does not incorporate predictive modeling of future workload states. Deng et al., (2016) similarly examined optimal workload allocation in fog-cloud computing with the goal of balancing delay and power consumption, reinforcing that the delay-energy trade-off is a recurring theme in this literature. Sartzetakis, Pantazopoulos, Katsaros, Sourlas, and Varvarigos (2024) extended this line of work to dynamic edge/cloud resource allocation under semi-static demand, underscoring that resource-allocation strategies must remain robust to demand patterns that vary over time. Complementing these systems-oriented studies, George et al. (2023) surveyed industry perspectives on the future of edge and cloud computing, Alsadie (2022) reviewed artificial-intelligence techniques for resource management in fog computing and categorized them by scalability, optimization potential, and flexibility across heterogeneous IoT environments, Aldossary (2021) reviewed dynamic resource-management techniques in cloud computing and emphasized the need for predictive, virtualization-aware provisioning, and Gadani (2024) examined the integration of edge computing with cloud services to improve application performance and reduce system overhead through efficient task offloading. Together these studies show that resource allocation in edge, fog environments is a well-known and multi-faceted research problem with very recent surveys continuously leading toward predictive, learning-based approaches as one of the means to complement purely optimization-based allocation schemes.

2.2 Machine Learning for Edge and Cloud Resource Prediction

The second stream of research applies classical and shallow machine–learning approaches to predict resource demand. For instance, Li et al. (2021) applied hybrid machine-learning methods such as Artificial Neural Networks (ANNs), yielding model-based time series forecasting and managerial insights in oil-and-gas field operations by enhancing energy-consumption efficiency, thus establishing the potential of an ANN to explain complex nonlinear energy-utilization patterns useful for downstream resource-management planning. To reduce overall resource cost and energy consumption in a cloud environment, Sangani et al. (2024) first proposed an error-optimization and resource-prediction algorithm to mitigate system statistics implications on prediction improvement due to explicit error-correction mechanisms being integrated with the basic architecture leading to putative improvements for individual processing units thus would improve the-quality of-resource-provisioning decisions. Using commonly applied architectures (CNN, LSTM, and Transformer models), Kamble et al. (2024) explored predictive resource-allocation strategies in cloud computing systems, concluding that the Transformer model performed best among those tested. Resource allocation is an important aspect of edge computing. Gunjal et al. (2024) proposed a dynamic resource-allocation approach that combines neural networks and decision-tree models to manage resources under variable workloads more quickly than existing machine-learning-based options using the described technique. Singh and Singh (2023) provided a survey on machine-learning techniques applied with fog- and cloud-computing domains for IoT-based robotics applications, emphasising the privacy-preserving communication mechanisms and smart resource-coordination approaches. In this set of studies, prediction accuracy and adaptability to variations in workload are the two evaluation criteria most often repeated and several authors explicitly state that improvements beyond what is achieved with deeper or single-model learners can be expected by using more temporally aware architectures, setting the stage for the deep-learning-based work reviewed next.

2.3 Deep Learning-Based Prediction and Hybrid Optimization Approaches

On top of the machine-learning literature, a series of recent articles combining deep-learning components with optimization algorithms. Gongada et al. (2024) proposed hybrid optimization-based deep-learning approaches that effectively address resource allocation under dynamic edge workloads by combining a convolutional neural network and a fruit-fly optimization algorithm to predict the future behavior of individual workload demands; with the results indicating accurate predictions of the load condition across various resources ranging from idle, moderate, to heavy. Sodiya et al. (2024) examined the current state and prospects of edge computing within the broader IoT ecosystem and highlighted computational complexity and hardware constraints as persistent barriers, concluding that prediction models must be refined to handle increasingly complex workload patterns. Abbas et al., (2018) surveyed mobile edge computing broadly and noted that deep-learning-based workload and resource-demand prediction is

promising for proactive resource management, but that the computational complexity of deep architectures typically necessitates pairing them with lightweight or optimized deployment strategies for resource-constrained edge nodes. Yuan et al. (2019) examined adversarial robustness in deep-learning systems more generally, which is relevant to this literature insofar as it underscores that deep forecasting models deployed in operational, safety- or cost-relevant settings such as edge resource management must also be evaluated for robustness, not accuracy alone. Most directly relevant to the attention-based design adopted in the present study, Guo et al. (2025) proposed a CNN-LSTM network based on a multi-modal attention mechanism for power-grid load forecasting and showed that incorporating attention over recurrent feature representations improves contextual learning and forecasting robustness for energy-related time series, providing direct empirical support for the use of attention mechanisms in the type of energy-forecasting problem addressed here.

2.4 LSTM-Based and Recurrent Sequence-Learning Approaches

Long Short-Term Memory (LSTM) networks and their variants form the recurrent backbone of much of the deep-learning-based forecasting literature reviewed above. Kamble et al. (2024) included LSTM as one of three deep architectures (alongside CNN and Transformer models) compared for cloud resource-allocation forecasting, and reported that recurrent and attention-based sequence models outperformed simpler baselines. Guo et al. (2025) likewise built on an LSTM backbone, augmenting it with convolutional feature extraction and multi-head attention for load forecasting. These studies collectively support the use of LSTM-family architectures and specifically their bidirectional variants, which process a sequence in both temporal directions as a strong foundation for resource and energy-consumption forecasting, while also indicating that unidirectional, single-branch LSTM models are increasingly being augmented rather than used in isolation, motivating the hybridization pursued in this paper.

2.5 GRU-Based and Hybrid Deep-Learning Architectures

A GRU network is a lighter-weight version of an LSTM: it combines the forget and input gates into a single update gate, while maintaining approximately comparable modeling power on sequences that are not too long, making training time faster and memory footprint smaller. Within the specific set of studies reviewed for this paper, GRU-based forecasting is comparatively under-represented relative to LSTM-based and CNN-based approaches this asymmetry is itself an observation worth noting rather than a gap to be papered over, since it indicates that the specific combination of bidirectional GRU with bidirectional LSTM rather than GRU alone is comparatively unexplored in the edge-resource-forecasting literature surveyed here. Hybrid architectures more broadly, such as the CNN-fruit-fly-optimization model of Gongada et al. (2024) and the CNN-LSTM-attention model of Guo et al. (2025), demonstrate that combining complementary components convolutional, recurrent, optimization-based, or attention-based tends to outperform any single component in isolation, which is the design principle extended in this paper to the specific combination of BiLSTM, BiGRU, multi-head self-attention, and residual learning.

2.6 Synthesis and Transition to the Research Gap

Taken together, the literature reviewed systems-level and optimization-based resource-allocation strategies, shallow machine-learning-based prediction deep-learning and hybrid optimization approaches with LSTM-family recurrent networks as the dominant sequence-learning backbone and GRU-based hybridization comparatively less explored. Several studies explicitly call for models that jointly capture short-term fluctuations and long-range dependencies while remaining computationally tractable for edge deployment, and the attention-augmented CNN-LSTM model of Guo et al. (2025) provides direct evidence that attention mechanisms improve contextual forecasting for energy-related time series specifically. Nevertheless, none of the examined studies unite both bidirectional LSTM + bidirectional GRU branches with multi-head self-attention and residual learning into a formally specified architecture evaluated on fine-grain edge/household energy-consumption data.

3. RESEARCH GAP, MOTIVATION, AND RESEARCH CONTRIBUTIONS

3.1 Research Gap

Edge computing enables data processing close to the source of origin and is key for lowering latency, bandwidth use, and energy consumption in intelligent distributed systems. Commonly, in energy and resource-demand prediction, auto regressive integrated moving average (ARIMA) based methods and linear regression based methods are used, which have limited applicability in such applications as they cannot capture the data behaviour due to dynamic temporal patterns that typically exhibit edge workloads. This limitation led to the subsequent proposal of machine-learning models over time, deep-learning models that helped improve prediction accuracy through learning hierarchical, temporally aware feature representations directly from historical data. Existing studies have primarily focused on either optimization-based allocation strategies, shallow machine-learning prediction, or single-branch deep-learning forecasting, while comparatively limited attention has been given to formally specified hybrid recurrent architectures that indicate

- Model bidirectional temporal context through complementary recurrent branches
- Apply attention mechanisms to weight informative time steps
- Incorporate residual learning to preserve robustness under sudden workload spikes all within a single framework evaluated specifically on fine-grained, edge-relevant energy-consumption data.

This is the unresolved problem the present study addresses an existing hybrid recurrent forecasting frameworks for edge and cloud resource management remain limited in their integration of short-term fluctuation modeling, long-term dependency capture, and contextual attention within one formally defined and empirically validated architecture, and this limitation constitutes the research gap that motivates the proposed Bi-LSTM-BiGRU hybrid framework.

3.2 Motivation for the Hybrid BiLSTM-BiGRU Design

The motivate for this hybrid design adopted are

- Resource and energy consumption time series at the edge exhibit both short-horizon fluctuations driven by transient device activity and longer-horizon periodic structure driven by daily and seasonal usage cycles, which favors an architecture that can represent both scales rather than one tuned to a single temporal resolution.
- Bidirectional recurrent processing used in both the BiLSTM and BiGRU branches allows the model to condition its representation of a given time step on both preceding and succeeding observations within the input window, which is appropriate in an offline forecasting-evaluation setting where the full historical window is available at prediction time.
- Pairing BiLSTM with BiGRU rather than deploying either alone reflects a deliberate accuracy/efficiency trade-off the BiLSTM branch's explicit cell-state memory is retained for its capacity to mitigate vanishing-gradient effects over longer sequences, while the BiGRU branch's simpler gating structure is retained for its comparatively lower parameter count and faster convergence, which is desirable in edge-oriented deployment scenarios where model retraining or fine-tuning may need to occur on constrained hardware.
- The multi-head self-attention layer and residual persistence connection are added on top of recurrent backbone specifically to address the two remaining weaknesses identified the absence of context-aware temporal weighting, and vulnerability to sudden workload spikes that a purely recurrent representation may fail to track without an explicit shortcut to the most recently observed value.

3.3 Research Contributions

To distinguish genuine architectural novelty from implementation and application contributions, the contributions of this study are stated explicitly below, with their nature indicated in parentheses.

- A formally specified, attention-enhanced hybrid recurrent architecture named Proposed-Mega that integrates one-dimensional convolutional feature extraction, recurrent processing, multi-head self-attention, and a persistence-based residual connection within a single trainable forecasting framework for dynamic edge environments (architectural contribution).
- An empirical evaluation of Proposed-Mega against four recurrent baselines—LSTM, BiLSTM, BiGRU, and a non-attention Hybrid model—using MAE, MSE, RMSE, R^2, and MAPE on both training and held-out test partitions, together with five-fold cross-validation of the proposed model's RMSE experimental contribution.
- A model-development comparison across the five evaluated architectures, examining training/validation behaviour and the relative performance of the recurrent and hybrid configurations, together with an explicit statistical-validation framework describing how repeated-run significance testing can be conducted in future extensions.
- An analysis connecting forecasting-accuracy differences among the compared architectures to their practical implications for proactive, prediction-driven resource management at the edge, including a candid discussion of the observed test-set performance and its limitations.

The application of established recurrent-network components is itself well established in the deep-learning literature; the contribution of this work lies specifically in their joint architectural integration for the edge-resource-forecasting problem, in the systematic empirical evaluation of that integration, and in transparent reporting of the resulting metrics, rather than in the invention of new recurrent-cell mathematics.

4. METHODOLOGY

4.1 Problem Formulation

Let $x_t \in \mathbb{R}^d$ denote the vector of d engineered features describing edge/household energy consumption and related sub-metering measurements observed at discrete time step t, and let a historical window of length L be denoted $X_{t-L+1:t} = (x_{t-L+1}, x_{t-L+2}, \ldots, x_t)$. The forecasting task addressed in this paper is to learn a function f_θ, parameterized by the trainable weights θ of the proposed hybrid network, that maps the historical window to a prediction of the target resource-utilization value at the next time step:

$$\hat{y}_{t+1} = f_\theta(X_{t-L+1:t}) \quad (1)$$

The parameters θ are learned by minimizing a supervised loss between $\hat{y}_{t+1}$ and the ground-truth observation y_{t+1} over the training partition. In the resource-allocation context that motivates this study, $\hat{y}_{t+1}$ is treated as a proxy for near-future resource/energy demand, which downstream allocation logic can use to proactively provision or de-provision edge resources ahead of anticipated demand changes.

4.2 System Overview

The overall system considered the standard edge-forecasting pipeline outlined in Figure 2 with sensors and edge devices stream raw consumption data, which is preprocessed and feature-engineered the resulting feature sequence is passed to the predictive model to produce a forecast of near-future resource utilization.

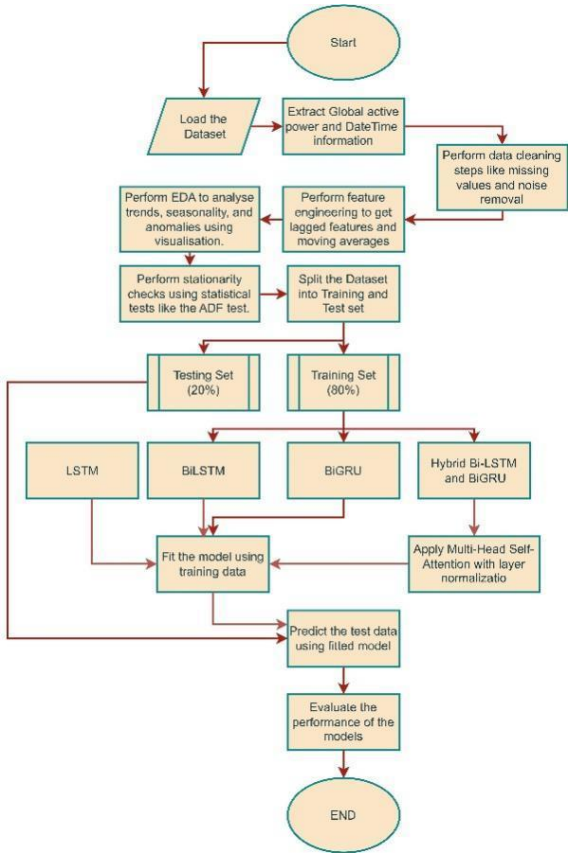


Figure 2. Training and Resource-Inference Workflow of the Proposed Framework

The forecast supports resource inferences such as detection of peak or moderate usage periods, anomaly identification, and load-management decisions; and the resulting insights are surfaced through a real-time dashboard while raw data and derived insights are stored locally and synchronized to the cloud for further analysis. This paper focuses on the modeling stage of the design, formal specification, and empirical evaluation of the predictive component while the dashboard, storage, and synchronization stages are described at the system level to situate the modeling contribution within a deployable edge-resource-management workflow, and are not themselves subject to the experimental evaluation.

4.3 Dataset and Data Collection

The study uses the Individual Household Electric Power Consumption dataset from the UCI Machine Learning Repository (Hebrail & Berard, 2009), which contains detailed electric-power-consumption measurements for a single household at a one-minute sampling rate over a period of nearly four years. The dataset provides high-resolution measurements of global active power, global reactive power, voltage, current intensity, and three sub-metering channels, and it is widely used as a benchmark for time-series forecasting and predictive-analytics research. While the underlying data originate from a household rather than a multi-tenant edge-computing cluster, the dataset is used here as a proxy for fine-grained, high-frequency resource/energy-utilization signals of the kind that an edge node would need to forecast in practice; this substitution is a deliberate and disclosed methodological choice rather than a claim that the dataset was collected from an edge-computing deployment.

4.4 Data Preprocessing

During preprocessing, 738 missing values were identified in the raw electricity-consumption recordings and were addressed using forward-fill and backward-fill imputation to preserve temporal continuity. The cleaned series contained 622,080 raw one-minute observations for the analysed period. Daily aggregation produced 432 daily observations, of which 430 remained after three-day centered smoothing; subsequent lag/rolling-feature construction removed additional warm-up rows, yielding 423 final feature-ready daily observations, consistent with the reported 243/90/90 train–validation–test split. These counts refer to distinct preprocessing stages rather than different descriptions of the same final subset. The final modelling dataset contains 17 recorded and engineered attributes, as described in the data-preparation workflow.

4.5 Feature Engineering

In feature engineering, the imputed time series is transformed into a feature set, which is meant to enhance prediction accuracy and stability of the resulting model. Four groups of engineered features were applied as follows:

- Time-based features - The timestamp was broken out into calendar attributes that is hour of day, day of week, month, and seasonal indicators to assist the model in identifying periodic and seasonal consumption patterns.
- Lag features - Historical daily target values were incorporated through one-day and seven-day lag

features, allowing the model to use recent and weekly temporal context in the forecasting task.

- Rolling statistics - A seven-day rolling mean and rolling standard deviation were calculated to represent short-term fluctuations and local variability, while an exponential moving average provided an additional smoothed representation.
- Derived features - Additional predictors included first differences and cyclical calendar encodings for hour, day of week, and month, together with a weekend indicator.

4.6 Formal Specification of the BiLSTM Component

A standard LSTM cell maintains a cell state c_t and a hidden state h_t that are updated at each time step through a forget gate, an input gate, and an output gate, which jointly regulate what information is discarded, added, and exposed at each step. Given input x_t and the previous hidden and cell states h_{t-1}, c_{t-1}, the LSTM update equations are:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (3)$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (5)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (6)$$

$$h_t = o_t \odot \tanh(c_t) \quad (7)$$

where f_t, i_t, and o_t are the forget, input, and output gate activations, $\tilde{c}_t$ is the candidate cell state, $\sigma(\cdot)$ is the logistic sigmoid function, $\odot$ denotes element-wise multiplication, and $W_{(\cdot)}$, $U_{(\cdot)}$, $b_{(\cdot)}$ are trainable weight matrices and bias vectors. A Bidirectional LSTM (BiLSTM) processes the input sequence with two independent LSTMs: a forward LSTM that computes $\rightarrow h_t$ from x_1 to x_L, and a backward LSTM that computes $\leftarrow h_t$ from x_L to x_1. The two hidden-state sequences are concatenated at each time step to form the BiLSTM output:

$$h_t^{BiLSTM} = [\rightarrow h_t \;;\; \leftarrow h_t] \quad (8)$$

This concatenation allows the representation at each time step to incorporate information from both past and future observations within the input window, which is appropriate in the offline forecasting-evaluation setting used in this study, where the complete historical window is available before the target step is predicted.

4.7 Formal Specification of the GRU Component

The GRU simplifies the LSTM cell by combining the forget and input gates into a single update gate and eliminating the separate cell state, which reduces the number of trainable parameters and typically shortens training time. Given input x_t and previous hidden state h_{t-1}, the GRU update equations are:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \quad (9)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \quad (10)$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \quad (11)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (12)$$

where z_t is the update gate, which controls how much of the previous hidden state is retained versus replaced by the candidate hidden state $\tilde{h}_t$, and r_t is the reset gate, which controls how much of the previous hidden state contributes to the candidate. As with the BiLSTM branch, a BiGRU computes forward and backward hidden-state sequences $\rightarrow h_t$ and $\leftarrow h_t$ using two independent GRUs traversing the sequence in opposite directions, and concatenates them:

$$h_t^{BiGRU} = [\rightarrow h_t \;;\; \leftarrow h_t] \quad (13)$$

4.8 Proposed Hybrid Attention-Residual Architecture (Proposed-Mega)

The proposed architecture, denoted Proposed-Mega, integrates the BiLSTM and BiGRU components specified above within a single forecasting pipeline consisting of five stages.

- Stage 1: Convolutional feature extraction - The input window X(t-L+1:t) is first passed through a one-dimensional convolutional layer, which extracts local short-term patterns and fluctuation trends from the sequential data before the recurrent processing stage.
- Stage 2: Hybrid bidirectional recurrent modelling - The convolutional representation is processed by a bidirectional LSTM layer to capture temporal context in both directions. The resulting sequence representation is then passed to the attention and bidirectional GRU stages, providing complementary recurrent modelling within the proposed hybrid architecture.
  $$h_t^{Hybrid} = [h_t^{BiLSTM} \;;\; h_t^{BiGRU}] \quad (14)$$
- Stage 3: Multi-head self-attention improve context understanding, a multi-head self-attention layer is applied over the hybrid recurrent representation h_t^{Hybrid} to focus on informative temporal regions and assign relevance weights to time steps that are most predictive of future resource demand. Each attention head applies scaled dot-product attention to a learned subspace of the hybrid representation:
  $$\text{head}_i = \text{softmax}((Q W_i^Q (K W_i^K)^T)/(\sqrt{(d_k)})) V W_i^V \quad (15)$$
  $$\text{MultiHead}(Q,K,V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h) W^O \quad (16)$$
  where Qi, Ki and Vi are the query, key and value representations obtained from h_tHybrid respectively; dk is the dimension of each head's query (used to scale the dot product for numerical

stability), h is the number of heads in the multiheaded attention module and W_iQ, W_iK, W_iV and WO are trainable projection matrices. The model perform the attendance over multiple aspects of the input sequence simultaneously, this mechanism is an adaptation of attention-based forecasting. This can improve both interpretability and robustness in respects to the generated forecasts for power-grid load prediction.

- Stage 4: A residual persistence connection along with normalization mitigate information loss and robustness to sudden spikes, which are common in edge-computing workloads where noise and bursty demand are prevalent. A persistence-based residual connection incorporates the most recent value of the target variable y_t directly into the representation feeding the output layer:
$z_t = \text{LayerNorm}(\text{Attn}(h_t\text{Hybrid}) + y_t)$ (17)
In this stage, layer normalization and dropout regularization in order to stabilise convergence, reduce overfitting and improve generalisation.
- Stage 5: The normalized, attention-weighted and residual-augmented representation z_t is fed through a dense fully connected layer to reach the predicted resource/energy-consumption value of interest:
$\hat{y}_{t+1} = W_{out}\, z_t + b_{out}$ (18)

4.9 Model Training

Network parameters θ across the five compared architectures are trained by minimizing a regression loss between predicted and actual target values over the training partition using gradient-based optimization. During training, dropout and, where applicable, layer normalization are used to stabilize convergence and reduce overfitting. Figure 7 presents the training and validation loss history for Proposed-Mega. The curves decrease rapidly during the early epochs and then stabilize, with no persistent divergence between training and validation loss that would by itself indicate severe overfitting.

4.10 Resource Allocation Strategy

The forecast $\hat{y}_{t+1}$ produced by the predictive model is intended to inform resource-allocation decisions in three complementary ways:

- Proactive provisioning- The anticipated increases in predicted demand trigger allocation of additional computational or bandwidth resources ahead of the demand materializing, rather than reactively after a threshold is breached
- Anomaly-aware load management- The large deviations between recent observed values and the model's expectation are flagged for operator or automated review, consistent with the anomaly-detection inference
- Energy-aware scheduling- in which periods of predicted low demand are used to schedule non-urgent workloads or defer resource-intensive tasks.

This study focuses on validating the predictive component underlying these strategies. The forecast is therefore evaluated as a predictive signal that could inform proactive provisioning, anomaly-aware monitoring, and energy-aware scheduling; a complete closed-loop resource-allocation policy is not implemented or experimentally evaluated in this work.

4.11 Prediction Process

At inference time, the trained model consumes the most recent feature window $X_{t-L+1:t}$, constructed using the same preprocessing and feature-engineering pipeline as during training and produces a one-step-ahead forecast $\hat{y}_{t+1}$ via the five-stage computation described that the forecast can be produced repeatedly as new observations arrive, supporting the real-time dashboard and resource-inference functions.

4.12 Evaluation Metrics

Model performance is evaluated using five complementary quantities: MAE, MSE, RMSE, R^2, and MAPE, computed on the training and test partitions. MAPE is reported as a percentage (%) throughout the results, consistent with equations.

$$MAE = (1)/(n)\Sigma_{i=1}^{n}|y_i - \hat{y}_i| \quad (19)$$

$$RMSE = \sqrt{((1)/(n)\Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2)} \quad (20)$$

$$R^2 = 1 - (\Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2)/(\Sigma_{i=1}^{n}(y_i - \bar{y})^2) \quad (21)$$

$$MAPE = (100\%)/(n)\Sigma_{i=1}^{n}|(y_i - \hat{y}_i)/(y_i)| \quad (22)$$

where y_i and $\hat{y}_i$ are the observed and predicted values for the i-th sample, n is the number of samples, and $\bar{y}$ is the mean of the observed values..

5. EXPERIMENTAL SETUP AND RESULTS

5.1 Tools and Platforms

Implementing and evaluating hybrid prediction models for edge computing requires a robust suite of platforms and libraries. Widely adopted deep-learning frameworks such as TensorFlow, PyTorch, and Keras provide the extensive libraries and pretrained-model support that simplify the construction and training of the recurrent and attention-based components and were drawn upon in implementing the compared architectures. These frameworks additionally offer edge-deployment services that integrate with their corresponding cloud counterparts, which is relevant to the eventual deployment of the proposed framework across heterogeneous edge environments.

5.2 Stationarity Analysis

Before model training, the statistical stationarity of the daily smoothed power-consumption series was assessed using the Augmented Dickey-Fuller (ADF) test:

$$ADF = (\hat{\gamma})/(SE(\hat{\gamma})) \quad (23)$$

where $\hat{\gamma}$ is the estimated coefficient on the lagged level term in the ADF regression and $SE(\hat{\gamma})$ is its standard error; the resulting statistic follows a non-standard distribution under the null hypothesis of a unit root, with critical values obtained from simulated distributions rather than the standard normal or t-distribution. The ADF statistic obtained for the processed series was -3.5755 with a corresponding p-value of 0.0062, which is below the conventional 0.05 significance threshold. The test statistic also exceeded the critical values at the 1% (-3.4463), 5% (-2.8686), and 10% (-2.5705) significance levels. These results indicate that the processed daily series does not contain a unit root and can be treated as stationary over the observed period, which supports the use of the recurrent architectures since stationarity is a standard precondition for stable learning and reliable generalization in sequence models trained on a single historical series.

5.3 Model Development and Validation Comparison

A model-development comparison was conducted across the five evaluated architectures to characterize their behaviour under the corresponding training/validation conditions. These results are intended to describe model development and should not be interpreted as a substitute for the final held-out test evaluation.

The values reported in Table 1 summarize development-stage training information, including stopping epoch, validation loss, training time, RMSE, and R^2. They are reported separately from the final held-out test metrics in Table 3 because the development and test evaluations serve different purposes.

Table 1. Model Development and Validation Summary

Model	Stopped epoch	val loss	time	RMSE	R^2
LSTM	26	0.11809	5.5s	0.2207	0.2256
BiLSTM	41	0.06990	12.7s	0.1551	0.6173
BiGRU	24	0.10827	8.3s	0.1772	0.5008
Hybrid	28	0.07686	41.1s	0.1550	0.6178
Proposed-Mega	35	0.13936	26.7s	0.1552	0.6170

Table 1 provides the development-stage comparison of the five models. BiLSTM and Hybrid show the lowest development RMSE (0.1551 and 0.1550, respectively), while Hybrid has the highest development R^2 (0.6178), closely followed by BiLSTM (0.6173) and Proposed-Mega (0.6170). Proposed-Mega has a development RMSE of 0.1552 and validation loss of 0.13936. These values are not used to rank the final held-out test performance.

5.4 Statistical Validation

The reported model comparison uses one train/test partition for each of the five architectures, together with five-fold cross-validation of RMSE for Proposed-Mega. Because the main comparison is based on a single held-out test partition, the reported differences should be interpreted as empirical performance differences rather than formal statistical-significance claims. The five-fold RMSE results for Proposed-Mega provide an additional assessment of performance stability across data partitions.

- For future work, repeated independent training runs can be performed for all five architectures, recording MAE, MSE, RMSE, R^2, and MAPE for each run.
- The resulting means and standard deviations can then be reported alongside the single-split results to quantify variability across independent runs.
- Paired statistical tests, such as a paired t-test when assumptions are satisfied or the Wilcoxon signed-rank test otherwise, can then be applied to per-run metric differences between Proposed-Mega and each baseline.
- When multiple pairwise comparisons are performed, an appropriate correction such as Holm-Bonferroni or Bonferroni should be applied before drawing conclusions about statistical significance.

5.5 Result Analysis

Accurate forecasting of power/resource consumption can support efficient resource management, reduced energy waste, and informed operational decision-making. The following analysis summarizes the temporal patterns and the final training and held-out test performance of the five evaluated architectures.

The monthly aggregated energy consumption shown in Figure 3 reaches its highest level in December and its lowest level in August. This pattern indicates variation across the months in the available household consumption record; because the analysis is based on a single household, the dataset alone does not support a causal explanation for the observed seasonal differences.

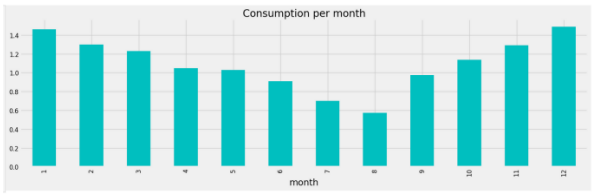


Figure 3. Monthly Energy Consumption per Month

Figure 4 presents an additive time-series decomposition of the consumption series into trend, seasonal, and

residual (noise) components. The decomposition separates the underlying trend from recurring seasonal structure and residual variation, providing complementary context for the stationarity and forecasting analysis.

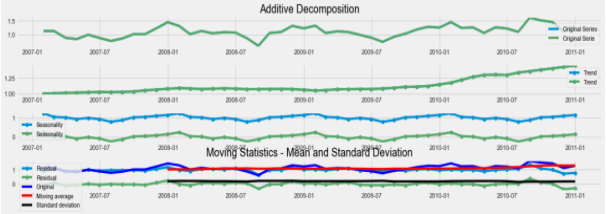


Figure 4. Trend and Seasonal Analysis of Additive Decomposition

5.5.1 Metric Consistency and Reporting Verification

A consistency audit was performed on the final training and test metrics to ensure that each reported quantity corresponds to the same evaluation convention. In particular, RMSE, R^2, and MAPE were checked using the final prediction and target arrays before reporting the results.

- For a common test target array, the identity $RMSE^2/(1-R^2)$ should equal the variance of the target series under the corresponding convention. For the five final test models, the calculated values are approximately 0.0629 for every model, confirming that the RMSE and R^2 values are mutually consistent to rounding precision.
- MAPE is reported as a percentage rather than a decimal ratio. Accordingly, the final test-set MAPE values are 14.2677% for LSTM, 11.7246% for BiLSTM, 13.9048% for BiGRU, 9.9566% for Hybrid, and 10.0245% for Proposed-Mega.
- These checks confirm that the final tables use a consistent target array and metric convention. The numerical results reported in Tables 2 and 3 are therefore treated as the authoritative training- and test-set results for the revised manuscript.

Table 2. Training Set Metrics Comparison

Model	MAE	MSE	RMSE	R2	MAPE
LSTM	0.0834	0.0108	0.1040	0.8469	8.2149
BiLSTM	0.1012	0.0161	0.1271	0.7716	10.3862
BiGRU	0.1061	0.0171	0.1308	0.7580	10.2694
Hybrid	0.0968	0.0153	0.1239	0.7829	9.8477
Proposed-Mega	0.0714	0.0092	0.0959	0.8699	7.1640

5.5.2 Training Set Performance

On the training partition, Proposed-Mega achieves the lowest MAE (0.0714), lowest MSE (0.0092), lowest RMSE (0.0959), highest R^2 (0.8699), and lowest MAPE (7.1640%). It is thertfore the strongest model on all five reported training metrics. Among the other models, LSTM has the next-highest R^2 (0.8469), while BiLSTM and Hybrid provide similar intermediate development and training behaviour.

5.5.3 Test Set Performance

On the held-out test set, Proposed-Mega achieves the lowest MAE (0.1137), lowest MSE (0.0198), lowest RMSE (0.1406), and highest R^2 (0.6855). Hybrid achieves the lowest MAPE (9.9566%), while Proposed-Mega records 10.0245%. Thus, Proposed-Mega does not lead every individual metric, but it provides the strongest overall balance across the five measures.

Table 3. Test Set Metrics Comparison

Model	MAE	MSE	RMSE	R2	MAPE
LSTM	0.1730	0.0494	0.2222	0.2146	14.2677
BiLSTM	0.1401	0.0308	0.1756	0.5098	11.7246
BiGRU	0.1671	0.0402	0.2005	0.3607	13.9048
Hybrid	0.1141	0.0212	0.1456	0.6630	9.9566
Proposed-Mega	0.1137	0.0198	0.1406	0.6855	10.0245

5.5.4 R^2 Comparison across Models

R^2 measures the proportion of variance in the observed target explained by the model, with higher values indicating better fit. Figure 5 shows that Proposed-Mega obtains the highest test-set R^2 (0.6855), followed by Hybrid (0.6630), BiLSTM (0.5098), BiGRU (0.3607), and LSTM (0.2146). The test-set R^2 range across the five models is therefore 0.2146–0.6855. The ordering is consistent with the overall ranking and indicates that the proposed architecture explains more of the observed test-set variation than the evaluated baselines.

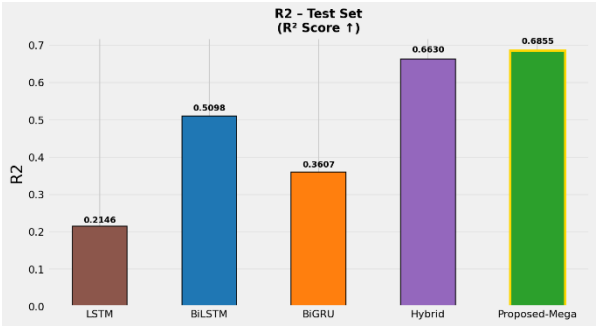


Figure 5. R^2 Comparison Across All Models (Test Set)

5.5.5 Cross-Validation RMSE of the Proposed Model

RMSE reduction indicates improved prediction accuracy and reduced forecast error. To assess the stability of Proposed-Mega beyond a single train/test split, five-fold cross-validation of RMSE was conducted. As shown in Figure 6, the per-fold RMSE values are 0.1744, 0.1775, 0.1679, 0.1749, and 0.2073 kW for folds 1–5, respectively. The mean RMSE is 0.1804 kW with a standard deviation of 0.0138 kW. The third fold has the lowest RMSE (0.1679 kW), while the fifth fold has the

highest (0.2073 kW). The spread provides evidence of moderate stability across partitions, while the higher fifth-fold error also shows that performance varies with the composition of the evaluation subset.

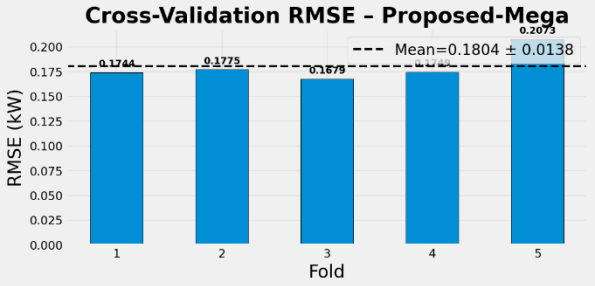


Figure 6. Cross-Validation RMSE of the Proposed Model

5.5.6 Learning Curve

Figure 7 shows the training and validation loss history for Proposed-Mega. Both curves decrease sharply during the early epochs and subsequently stabilize, and the best validation epoch indicated in the figure is epoch 52. The curves remain relatively close for most of the later training period, with no sustained separation that would by itself indicate severe overfitting. Because the figure reports the loss trajectory rather than a single summary statistic, no unsupported final loss values are assigned to the curves in the text.

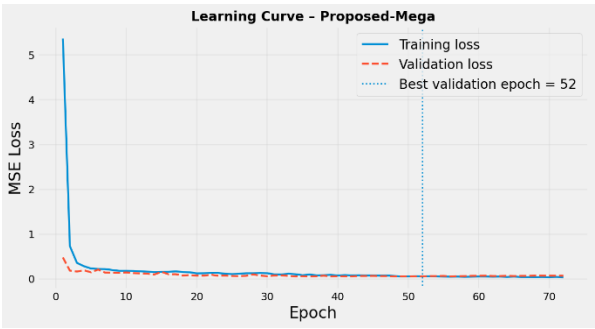


Figure 7. Proposed-Mega Training and Validation Learning Curve

5.5.7 Comparative Radar Analysis

Figure 8 presents the normalized test-set performance of the five models across RMSE, MSE, MAE, MAPE, and R^2, with the outer edge representing the better direction for each metric. Proposed-Mega and Hybrid occupy the strongest regions of the radar chart, while BiGRU and LSTM are generally farther from the preferred boundary. The two strongest models are close on several axes, with Hybrid having a slight advantage in MAPE and Proposed-Mega leading the other principal error and fit metrics.

The radar analysis is consistent with the numerical results: Proposed-Mega leads on MAE, MSE, RMSE, and R^2, whereas Hybrid has the lowest MAPE. This complementary pattern explains why the final overall ranking is based on all five metrics rather than on a single measure.

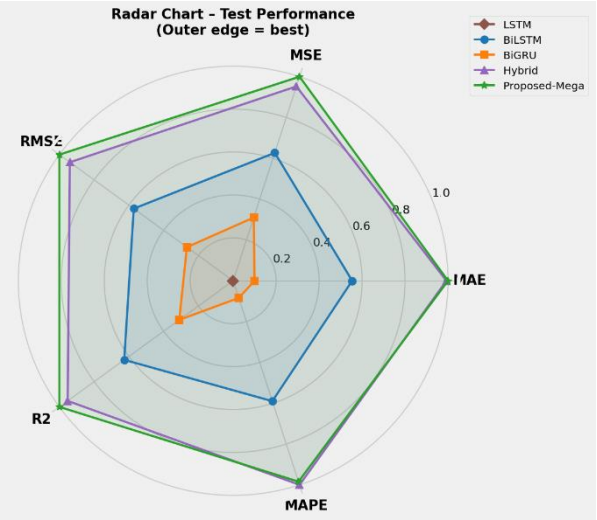


Figure 8. Radar Chart of Test-Set Performance Across All Models

6. DISCUSSION

6.1 Main Findings

The final test-set results show that Proposed-Mega provides the strongest overall predictive performance among the five evaluated architectures. It achieves MAE = 0.1137, MSE = 0.0198, RMSE = 0.1406, and R^2 = 0.6855. Hybrid achieves the lowest MAPE at 9.9566%, while Proposed-Mega records 10.0245%. Thus, Proposed-Mega leads four of the five individual metrics and provides the best aggregate performance when all five metrics are considered together.

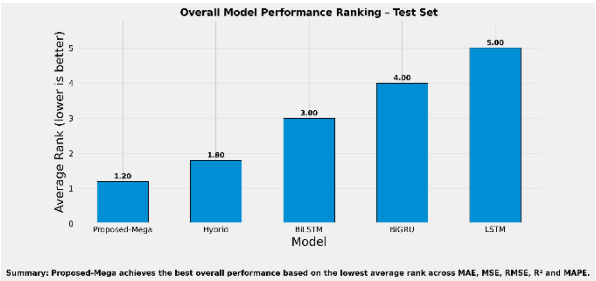


Figure 9. Overall Model Performance Ranking on the Test Set

6.2 Model Behavior

The final results show a difference between development-stage and held-out test performance. In the development comparison reported in Table 1, Proposed-Mega records RMSE = 0.1552 and R^2 = 0.6170. On the held-out test set, it records a lower RMSE of 0.1406 and a higher R^2 of 0.6855. The training metrics in Table 2 also show strong performance for Proposed-Mega, with the lowest MAE, MSE, and RMSE and the highest R^2 and lowest MAPE among the five models. Figure 7 shows

rapid early optimization followed by stabilization. These observations are consistent with good generalization on the present split, although the single-split design does not by itself establish a causal explanation for the train/test difference.

6.3 Comparison with Existing Research

The finding that the hybrid configurations outperform the single-branch recurrent baselines on most test metrics is broadly consistent with the hybrid-modeling literature reviewed in this paper. The additional use of multi-head self-attention in Proposed-Mega is associated with the strongest aggregate test ranking in the present experiment. However, comparisons with published results remain directional because the datasets, temporal resolutions, preprocessing procedures, and evaluation protocols differ. The R^2 range has been recalculated from the final held-out test results. The five test-set R^2 values are 0.2146 (LSTM), 0.5098 (BiLSTM), 0.3607 (BiGRU), 0.6630 (Hybrid), and 0.6855 (Proposed-Mega), giving a corrected range of 0.2146–0.6855 and should therefore not be interpreted as a like-for-like benchmark against larger grid-level forecasting studies.

6.4 Resource Allocation Implications

The forecasting results have practical relevance for resource-management strategies because the gap between predicted and actual demand can indicate potential over- or under-provisioning. Lower MAE and RMSE on unseen data can support more accurate proactive provisioning, while the higher test-set R^2 of Proposed-Mega (0.6855) indicates that it explains more of the observed test-set variation than the other evaluated models. These findings provide a predictive basis for resource-allocation decisions, but the study does not implement or evaluate a full closed-loop allocation policy. Any deployment decision should also account for safety margins, latency, computational cost, and model behaviour under atypical workloads.

6.5 Unexpected Results

An important result is that Proposed-Mega does not achieve the lowest value on every individual metric: Hybrid has a slightly lower MAPE (9.9566% versus 10.0245%). Nevertheless, Proposed-Mega ranks first overall because it leads on MAE, MSE, RMSE, and R^2, producing an average rank of 1.20 compared with 1.80 for Hybrid. This result demonstrates why the final comparison uses a multi-metric overall ranking rather than relying on a single error measure.

6.6 Statistical Reliability

The main model comparison is based on a single train/test split for all five architectures. Five-fold cross-validation was also conducted for Proposed-Mega using RMSE, producing a mean of 0.1804 kW and a standard deviation of 0.0138 kW across the five folds. This provides evidence of moderate RMSE stability for Proposed-Mega across partitions, but it does not provide repeated-run uncertainty estimates for all metrics and all models. Accordingly, the reported model differences should be interpreted as results from the present experimental split rather than as formally statistically significant differences.

6.7 Practical Implications

For practical edge deployments, the results suggest that an attention-enhanced hybrid architecture can provide a strong predictive basis when sufficient computational resources are available. Proposed-Mega offers the best aggregate test ranking, whereas Hybrid provides a slightly lower MAPE and may therefore be attractive when percentage error is the primary operational criterion. In either case, deployment should evaluate latency, memory, energy cost, and robustness under the target workload.

- The workload signal exhibits both short-term fluctuation and longer-range periodicity, as reflected in the temporal analysis of the household consumption record.
- The proposed architecture requires additional convolutional, recurrent, attention, normalization, and residual components; therefore, its computational cost should be considered when targeting resource-constrained edge hardware.

Where computational budget is tightly constrained, the non-attention Hybrid configuration may represent a reasonable accuracy/efficiency alternative, particularly when minimizing MAPE is the primary objective. This trade-off should be evaluated explicitly against the latency, memory, and energy budget of the target deployment.

6.8 Limitations

The following limitations should be considered when interpreting the results:

- The dataset used for evaluation is a single-household electric-power-consumption record rather than data collected from a multi-tenant edge-computing cluster. Although it serves as a proxy for fine-grained resource-utilization signals, transfer to genuinely multi-tenant, multi-resource-type edge workloads has not been evaluated here.
- The processed dataset comprises 423 feature-ready daily observations, which is a modest sample size for deep-learning evaluation and limits the statistical power of a single train/test comparison.
- The main model comparison uses a single train/test split, and only Proposed-Mega received five-fold RMSE cross-validation. Repeated multi-run evaluation of all models and metrics would

provide stronger estimates of variability and statistical reliability.

- This study assesses the predictive component of the proposed resource-allocation pipeline and does not implement or evaluate a full closed-loop resource-allocation policy driven by the forecasts.

7. CONCLUSION

The Proposed-Mega framework is an attention-enhanced residual hybrid forecasting architecture designed for resource and energy prediction in dynamic edge-computing environments. On the held-out test set, it achieves MAE = 0.1137, MSE = 0.0198, RMSE = 0.1406, and R^2 = 0.6855, while Hybrid achieves the lowest MAPE at 9.9566% compared with 10.0245% for Proposed-Mega. Considering all five metrics jointly, Proposed-Mega obtains the best overall average rank (1.20), followed by Hybrid (1.80), BiLSTM (3.00), BiGRU (4.00), and LSTM (5.00). These results indicate that the proposed architecture provides the strongest aggregate predictive performance on the evaluated dataset, while the slight MAPE advantage of Hybrid highlights the importance of metric-specific considerations.

Future work should strengthen statistical validation through repeated independent training runs and evaluate the framework across diverse datasets and realistic edge scenarios. Such validation should also examine computational cost, latency, and energy efficiency, and should test whether the predictive improvements translate into measurable gains when integrated with a genuine closed-loop resource-allocation policy.

References:

Abbas, N., Zhang, Y., Taherkordi, A., & Skeie, T. (2018). Mobile edge computing: A survey. *IEEE Internet of Things Journal, 5*(1), 450–465. https://doi.org/10.1109/JIOT.2017.2750180.

Aldossary, M. (2021). A review of dynamic resource management in cloud computing environments. *Computer Systems Science and Engineering*, *36*(3), 461–476. https://doi.org/10.32604/csse.2021.014975.

Alsadie, D. (2022). Resource management strategies in fog computing environment: A comprehensive review. *International Journal of Computer Science and Network Security*, *22*(4), 310-328. https://doi.org/10.22937/IJCSNS.2022.22.4.38

Dab, B., Aitsaadi, N., & Langar, R. (2019). A novel joint offloading and resource allocation scheme for mobile edge computing. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)* (Article 8651879). IEEE. https://doi.org/10.1109/CCNC.2019.8651879.

Deng, R., Lu, R., Lai, C., Luan, T. H., & Liang, H. (2016). Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption. *IEEE Internet of Things Journal, 3*(6), 1171–1181. https://doi.org/10.1109/JIOT.2016.2565516.

Gadani, N. N. (2024). Enhancing software development: Integrating edge computing with cloud services. *International Journal of Science and Research, 13*(8), 1394–1401. https://doi.org/10.21275/SR24822092829.

George, A. S., George, A. S. H., & Baskar, T. (2023). Edge computing and the future of cloud computing: A survey of industry perspectives and predictions. *Partners Universal International Research Journal*, *2*(2), 19-44. https://doi.org/10.5281/zenodo.8020101

Gongada, T. N., Desale, G. B., Ghodake, S. P., Sridharan, K., Rao, V. S., & El-Ebiary, Y. A. B. (2024). Optimizing resource allocation in cloud environments using fruit fly optimization and convolutional neural networks. *International Journal of Advanced Computer Science and Applications*, *15*(5). https://doi.org/10.14569/IJACSA.2024.01505119

Gunjal, A., Pednekar, R., Kanere, S., & Ruke, S. (2024). Dynamic resource allocation in edge computing environments: A machine-learning approach. *Journal of Trends and Challenges in Artificial Intelligence*, *1*(4), 133-138. https://doi.org/10.61552/JAI.2024.04.003

Guo, W., Liu, S., Weng, L., & Liang, X. (2025). Power-grid load forecasting using a CNN-LSTM network based on a multi-modal attention mechanism. *Applied Sciences*, *15*(5), 2435. https://doi.org/10.3390/app15052435

Hebrail, G., & Berard, A. (2009). *Individual household electric power consumption* [Data set]. UCI Machine Learning Repository. https://doi.org/10.24432/C58K54

Kamble, T., Deokar, S., Wadne, V. S., Gadekar, D. P., Vanjari, H. B., & Mange, P. (2024). Predictive resource allocation strategies for cloud computing environments using machine learning. *Journal of Electrical Systems*, *19*(2), 68-77. https://doi.org/10.52783/jes.692

Li, J., Guo, Y., Zhang, X., & Fu, Z. (2021). Using hybrid machine learning methods to predict and improve the energy consumption efficiency in oil and gas fields. *Mobile Information Systems*, *2021*, Article 5729630. https://doi.org/10.1155/2021/5729630

Mach, P., & Becvar, Z. (2017). Mobile edge computing: A survey on architecture and computation offloading. *IEEE Communications Surveys & Tutorials*, *19*(3), 1628-1656. https://doi.org/10.1109/COMST.2017.2682318

Mao, Y., You, C., Zhang, J., Huang, K., & Letaief, K. B. (2017). A survey on mobile edge computing: The communication perspective. *IEEE Communications Surveys & Tutorials*, *19*(4), 2322-2358. https://doi.org/10.1109/COMST.2017.2745201

Sangani, S., Patil, R., & Goudar, R. H. (2024). Efficient algorithm for error optimization and resource prediction to mitigate cost and energy consumption in a cloud environment. *International Journal of Information Technology*, *16*(4), 2187-2197. https://doi.org/10.1007/s41870-024-01732-1

Sartzetakis, I., Pantazopoulos, P., Katsaros, K. V., Sourlas, V., & Varvarigos, E. (2024). Dynamic edge/cloud resource allocation for distributed computation under semi-static demands. In *Proceedings of the IEEE International Conference on Communications (ICC)* (pp. 4024-4029). IEEE. https://doi.org/10.1109/ICC51166.2024.10622236

Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, *50*(1), 30-39. https://doi.org/10.1109/MC.2017.9

Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. *IEEE Internet of Things Journal*, *3*(5), 637-646. https://doi.org/10.1109/JIOT.2016.2579198

Singh, K. D., & Singh, P. D. (2023). Machine learning in robotics with fog/cloud computing and IoT. *EAI Endorsed Transactions on AI and Robotics*, *2*(1), Article e1. https://doi.org/10.4108/airo.3621

Sodiya, E. O., Umoga, U. J., Obaigbena, A., Jacks, B. S., Ugwuanyi, E. D., Daraojimba, A. I., & Lottu, O. A. (2024). Current state and prospects of edge computing within the Internet of Things (IoT) ecosystem. *International Journal of Science and Research Archive*, *11*(1), 1863-1873. https://doi.org/10.30574/ijsra.2024.11.1.0287

Yu, W., Liang, F., He, X., Hatcher, W. G., Lu, C., Lin, J., & Yang, X. (2018). A survey on the edge computing for the Internet of Things. *IEEE Access*, *6*, 6900-6919. https://doi.org/10.1109/ACCESS.2017.2778504

Yuan, X., He, P., Zhu, Q., & Li, X. (2019). Adversarial examples: Attacks and defenses for deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(9), 2805-2824. https://doi.org/10.1109/TNNLS.2018.2886017

Shreya Agarwal
Banasthali Vidyapith,
Jaipur,
India
shreya.gupta1606@gmail.com
0009-0000-9844-247X

Sonam Seth
Banasthali Vidyapith,
Jaipur,
India
sonamseth@banasthali.in
0000-0002-5898-0051