JOURNAL OF RENEWABLE ENERGY AND SMART GRID TECHNOLOGY

Vol. 20, No. 2, July-December 2025

# Hybrid Deep Learning Models for Energy Consumption Forecasting: A CNN-LSTM Approach for Large-Scale Datasets

Sri Harish Nandigam[1], K. Nageswararao[1*], Purnima K Sharma[2]

[1]Department of Electrical & Electronics Engineering, Hindustan Institute of Technology and Science, Chennai 603103, India
[2]Chitkara Institute of Engineering and Technology, Chitkara University, Punjab 140401, India

***Corresponding author's email**: nageswarak@hindustanuniv.ac.in

**Article info:**

Received: 26 March 2025
Revised: 2 June 2025
Accepted: 19 June 2025
DOI:
10.69650/rast.2025.261326

**Keywords:**

Electricity Consumption Forecasting
Smart Grids
Deep Learning
Hybrid CNN-LSTM Model
Energy Management

**ABSTRACT**

Long-term electricity consumption forecasting is essential with the rise of smart grids and advanced metering infrastructures for optimized energy management. Categorizing energy consumption into historical time series enables accurate predictions and deeper insights into energy trends. Deep learning has driven innovative forecasting models, particularly in smart grids leveraging data-driven approaches. This study applies convolutional neural networks (CNNs), gated recurrent units (GRUs), and long short-term memory (LSTM) networks to analyze complex energy consumption patterns. Hybrid CNN-GRU and CNN-LSTM models are proposed to enhance forecasting accuracy by capturing spatial and temporal correlations. A comparative analysis is conducted against standalone GRU and LSTM models using historical data from American Electric Power (AEP) and Dominion Virginia Power (DOM). Performance is assessed using symmetric Mean Absolute Percentage Error (sMAPE), Loss, and Root Mean Square Error (RMSE). The findings demonstrate that hybridCNN-LSTMmodels improve forecasting accuracy, enabling proactive strategies like load shedding for efficient energy management as compared to other models. This research contributes to both academia and industry by enhancing smart grid reliability and operational efficiency.

## 1. Introduction

Over the past two decades, global electricity consumption has observed a dramatic surge, primarily fueled by rapid economic development and exponential population growth. As economies expand and populations increase, the demand for energy escalates, making it a cornerstone of social and economic progress. Energy is not merely a resource but a critical enabler of development, influencing everything from industrial productivity to household well-being. It plays a focal role in shaping the trajectory of a region's growth, ensuring economic stability, and improving the quality of life for its inhabitants. Consequently, accurate forecasting of electricity consumption has become indispensable for effective energy supply management, infrastructure planning, revenue projection, capital investment decisions, and market analysis. Reliable predictions enable policymakers and energy providers to optimize resource allocation, reduce wastage, and ensure a stable energy supply, which is essential for sustaining economic and social development. However, despite its importance, long-term electricity consumption forecasting remains a formidable challenge. This complexity arises from the numerous uncertainties inherent in energy systems, such as fluctuating economic conditions, technological advancements, policy changes, and environmental factors. These uncertainties have historically limited scientific interest in long-term forecasting, prompting researchers to focus on developing more precise and reliable predictive models.

Energy consumption prediction is inherently a time-series problem, requiring the analysis of univariate or multivariate data collected from smart sensors and IoT devices. These datasets, while rich in information, often present significant challenges, including redundancies, missing values, outliers, and uncertainties. Additionally, seasonal variations and irregular trends in energy usage further complicate the forecasting process. Traditional machine learning models, such as linear regression or decision trees, often struggle to effectively capture the sequential patterns and nonlinear relationships present in energy data, leading to less accurate and unreliable forecasts. In contrast to these conventional approaches, deep learning models have demonstrated remarkable accuracy and versatility across a wide range of domains, including video summarization, image classification, and action recognition. Their ability to automatically learn complex patterns and relationships from large datasets has made them particularly well-suited for time-series forecasting tasks. In recent years, there has been a growing emphasis on integrating multiple deep learning models to enhance predictive performance. Hybrid models, which combine the strengths of different architectures, have achieved state-of-the-art results in energy forecasting [31]. For instance, models that integrate Convolutional Neural Networks (CNNs) for spatial feature extraction and Recurrent Neural Networks (RNNs) such as Long Short-Term Memory (LSTM) for temporal dependency modeling have demonstrated great promise.

These hybrid techniques are particularly effective in capturing both the geographical and temporal dynamics of energy consumption data, resulting in more accurate and robust forecasts. Despite their success, hybrid deep learning models have limits. One of the key issues is their computational inefficiency, which frequently leads to longer training times and increased resource requirements. Additionally, short-term forecasting delays can occur due to the complexity of these models, making them less practical for real-time applications. These limitations have spurred ongoing research into optimizing hybrid models for better computational efficiency and faster inference times, ensuring their applicability in real-world scenarios. Researchers are also exploring techniques such as model pruning, quantization, and parallel processing to address these challenges and make hybrid models more accessible for practical energy forecasting tasks. In summary, while the rise in global electricity consumption has underscored the need for accurate forecasting, the inherent complexities of energy systems have made this task increasingly challenging. Traditional machine learning models, though useful, often fall short in capturing the intricate patterns and dependencies in energy data. Deep learning models, particularly hybrid architectures, have emerged as powerful tools for addressing these challenges, offering superior accuracy and performance. However, their computational demands and practical limitations highlight the need for continued innovation and optimization. By advancing these models, researchers aim to develop more efficient and reliable forecasting tools that can support sustainable energy management and contribute to global economic and social development.

## 2. Literature Review

The hybrid CNN-LSTM model has become a powerful way to improve the accuracy of forecasts by capturing both spatial and temporal patterns in data, which is something that traditional forecasting methods often fail to do. This model combines the best features of Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks. CNNs are great at finding spatial features in data, while LSTMs are great at showing how things change over time. The hybrid model works very well in many different situations because it can do two things at once. It can handle complex, nonlinear, and time-varying datasets. For instance, when it came to estimating global horizontal irradiance (GHI), the hybrid CNN-LSTM-GRU model did better than standalone models, with lower mean absolute error (MAE) and root mean squared error (RMSE). The result highlights its ability to minimize prediction errors and improve accuracy, particularly in energy-related forecasting tasks [1]. LSTM networks and hybrid CNN-LSTM architectures have been the main tools used in recent energy forecasting advances to understand temporal dependencies in electricity usage. Despite their success in capturing nonlinear consumption patterns, these models nevertheless have a number of drawbacks. Few studies examine utility-specific load forecasting, particularly when using real consumption data like AEP and DOM, and most rely on public or synthetic datasets. Additionally, there aren't many studies that compare deep learning models with statistical baselines (like ARIMA) or use lightweight RNN variations (like GRUs) to cut down on training overhead [32]. In the same way, the hybrid Transformer-LSTM model did better at predicting glucose levels than standard LSTM models, with lower mean square error (MSE) values. This achievement underscores its effectiveness in managing the nonlinear and dynamic nature of glucose data, which is critical for healthcare applications [2].

The CNN-Bi-LSTM model, which was improved using genetic algorithms, had an amazing accuracy rate of 99.4% when predicting the intensity of cyclones, which was a lot better than other models. This success is attributed to the model's ability to capture intricate spatial-temporal patterns, which are essential for accurate weather predictions [3]. Also, the CNN-LSTM model combined with variational mode decomposition made more accurate predictions about photovoltaic power than other deep learning models. This result demonstrates its robustness and reliability in energy management systems, where precise forecasting is crucial for optimizing power generation and distribution [4]. The model's use in predicting influenza activity also showed how well it could help prevent disease by correctly predicting the rates of influenza-like illnesses, which is important for planning public health [5]. The CNN-LSTM model along with Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) cut RMSE by 58% compared to benchmark models when it came to predicting wind speed. This result highlights its proficiency in handling nonlinear variations in wind speed data, which is critical for renewable energy applications [6]. Also, when it came to predicting retail sales, hybrid CNN-LSTM models did better than traditional autoregressive methods because they were easier to use and required less computing power. This advantage makes them highly suitable for dynamic and data-intensive industries [7]. Other domains have also demonstrated the ability of the hybrid CNN-LSTM model to integrate spatial and temporal data processing. For example, the model did better than traditional methods at predicting stock prices because it was better at dealing with long-term dependencies and getting useful features from the data, which are important for making accurate predictions in markets that change a lot [1, 7]. When it comes to weather forecasting, the CNN-LSTM model was more accurate and stable than older physical and statistical models, with lower MSE and RMSE values. Such behavior indicates its enhanced predictive capabilities and its ability to process complex meteorological data, aligning prediction curves closely with test data [2] [4]. The hybrid model, which uses CNN for feature extraction and LSTM for temporal modeling, was more accurate than non-parametric models at predicting traffic flow [6, 8]. This shows how well it works in real-world situations. The hybrid CNN-LSTM model works well because it preprocesses a lot of data, tunes a lot of parameters, and combines CNN and LSTM architectures. These elements collectively enhance its ability to learn from and predict complex patterns in time series data [1][10]. The hybrid model is a big step forward in predictive analytics because it fixes problems with traditional methods, like the fact that they can't show nonlinear relationships and sequential dependencies. Its versatility and accuracy make it a valuable tool for various forecasting tasks, including energy management, healthcare, weather prediction, and financial markets. Overall, the hybrid CNN-LSTM model is a huge step forward in forecasting technology. It makes accurate and reliable predictions that help people make decisions and improve things in many areas.

## 3. Proposed electricity consumption prediction framework:

Predicting electricity use accurately increases energy efficiency and empowers building managers to make informed energy management decisions, resulting in significant energy savings. However, precisely predicting energy use is a difficult task because interruptions are random and unpredictable. To guarantee reliable outcomes in energy consumption forecasting, this research presents a novel framework, which is illustrated in Figure 1 and elaborated upon in the subsequent section. Three essential components make up this framework: (1) preprocessing of the data, (2) feature extraction, and (3) assessment and forecasting.

This process includes cleaning the raw data to get rid of errors, noise, and other oddities, standardizing it to make sure that all the features are scaled the same, and filling in missing data using methods like interpolation or imputation.

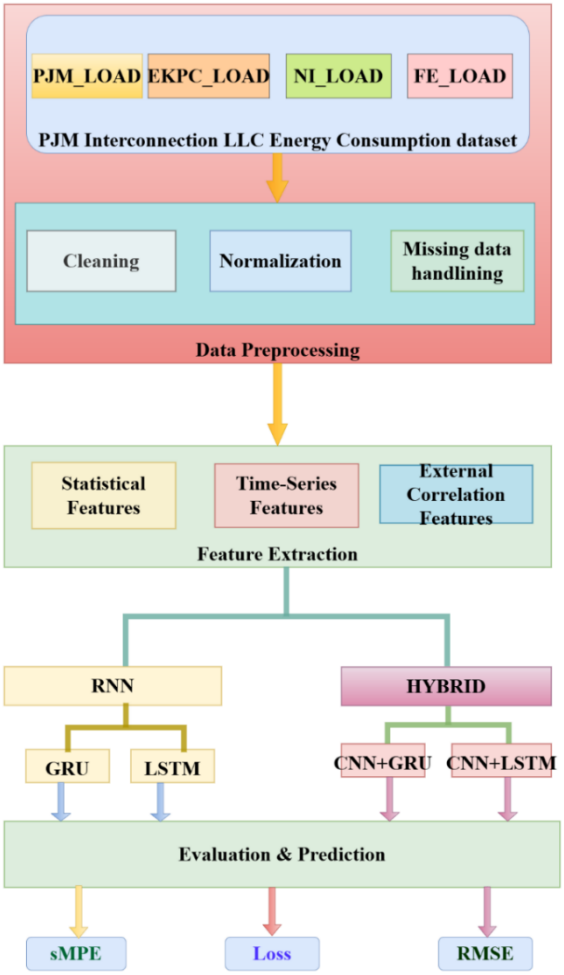


Fig.1 Conventional and hybrid model frame work for long-term energy consumption prediction.

Following the preprocessing phase, the data progresses to the feature extraction stage, wherein significant patterns are discerned. Statistical features (like mean, variance, and standard deviation), time-series features (like trends and seasonal changes), and external correlation features that show links between consumption and environmental or contextual variables are all part of this process. After the feature extraction step, predictive modeling is done using two frameworks: traditional deep learning methods, like GRU (Gated Recurrent Unit) and LSTM (Long Short-Term Memory), which are great at working with sequential and time-series datasets; and hybrid deep learning methods, like CNN-GRU and CNN-LSTM, which combine the feature extraction strengths of convolutional neural networks with the temporal processing strengths of recurrent neural networks. Finally, the models are tested and proven using sMPE (Symmetric Mean Percentage Error) to see how accurate the predictions are, loss to see how well the training worked, and RMSE (Root Mean Square Error) to see how big the errors are in the predictions. Using advanced machine learning techniques and a structured data management method, this all-encompassing framework ensures accurate and reliable predictions of electricity use.

*3.1 Data Preprocessing and Sliding Window Implementation for Electricity Consumption Forecasting*

Accurate electricity consumption forecasting is crucial for effective energy management and optimization. This study presents a structured data preprocessing pipeline that extracts relevant features, scales the data, and utilizes a sliding window technique for time-series modeling. The methodology ensures proper data segmentation for training, validation, and testing.

**Defining Features and Target Variable**

**Label Column:** The target variable is defined as electricity consumption (index 0), which is the value to be predicted.

**Input Features:** The first five columns are selected as inputs, including consumption, hour of the day, day of the week, month, and day of the year:

$$\boldsymbol{input\_cols\_indices} = \boldsymbol{range}(5) = (\boldsymbol{consumption}, \boldsymbol{hour}, \boldsymbol{day\ of\ week}, \boldsymbol{month}, \boldsymbol{day\ of\ year})$$

**Sliding Window Size:** A window size of 90 is chosen, meaning each input sequence consists of 90 consecutive time steps:

$$window\_size = 90$$

**Data Storage Initialization:** To store the processed data, the following structures are initialized:

**label_scalers**: Dictionary to store MinMaxScaler instances for inverse transformation.

**Training Data:** train_features, train_labels

**Validation Data:** val_features, val_labels

**Testing Data:** test_x, test_y, stored separately for each dataset file.

*3.2 File Processing and Feature Extraction*

Each CSV file is read, parsed, and time-related features are extracted:

```
df = pd.read_csv(os.path.join(data_dir, "file"),
    parse_dates=["Datetime"])
```

The following time-based features are computed:

Hour of the day: $h_t = \text{hour}(t)$

Day of the week: $d_t = \text{dayofweek}(t)$

Month of the year: $m_t = \text{month}(t)$

Day of the year: $d_{yt} = \text{dayofyear}(t)$

*3.2.1 Data Sorting and Normalization*

To ensure chronological ordering, the dataset is sorted and the Datetime column is dropped:

df = df.sort_values("Datetime").drop("Datetime", axis=1)

The data is scaled using MinMaxScaler:

sc = MinMaxScaler() label sc = MinMaxScaler()

Each feature is transformed:

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

The target variable (electricity consumption) is separately scaled:

$$Y' = \frac{Y - Y_{min}}{Y_{max} - Y_{min}}$$

*3.2.2 Creating Sliding Window Sequences*

A function move_sliding_window() generates sequential input-label pairs:

$$X_T = \{x_t - W + 1, x_t - W + 2, \ldots., x_t\}$$

$$Y_t = y_t + 1$$

Each sequence consists of 90 data points (as per the window size), with the label being the predicted consumption value.

*3.2.3 Data Splitting Strategy*

The dataset is divided into training (80%), validation (10%), and testing (10%) sets:

Total samples: $N = \boldsymbol{len(inputs)}$

Split portions: $\boldsymbol{N_{test}} = [0.05 \times N]$, $\boldsymbol{N_{val}} = [0.05 \times \mathrm{N}]$

Training end index: $\boldsymbol{N_{train}} = \boldsymbol{N} - (\boldsymbol{N_{test}} + \boldsymbol{N_{val}})$

*3.2.4 Assigning Data Splits*

Training Set: $\boldsymbol{X_{train}} = \boldsymbol{X}[:\boldsymbol{N_{train}}]$,

$\boldsymbol{Y_{train}} = \boldsymbol{Y}[:\boldsymbol{N_{train}}]$

$\mathrm{Y_{train}} = \mathrm{Y}[:\mathrm{N_{train}}]$Validation

$\boldsymbol{Y_{val}} = \boldsymbol{Y}[\boldsymbol{N_{train}}:\boldsymbol{N_{train}} + \boldsymbol{N_{val}}]$

Testing Set: $\boldsymbol{X_{test}} = \boldsymbol{X}[-\boldsymbol{N_{test}}:]$,

$\boldsymbol{Y_{test}} = \boldsymbol{Y}[-\boldsymbol{N_{test}}:]$

*3.2.5 Data Aggregation across Multiple Files*

To enhance the dataset, training and validation data from multiple files are merged:

$$X_{train}^{final} = \bigcup X_{train}^{files}, Y_{train}^{final} = \bigcup Y_{train}^{files},$$

The test data remains separate for each file:

$$X_{test}[file] = X_{train}^{file}, Y_{test}[file] = Y_{test}^{file}$$

This structured approach enables efficient time-series modeling of electricity consumption data. The combination of feature engineering, normalization, and a sliding window technique allows the model to effectively capture sequential patterns, leading to improved forecasting accuracy.

*3.3 Evaluation and prediction*

*3.3.1 Conventional GRU (Gated Recurrent Unit)*

Figure 2 represents a GRU (Gated Recurrent Unit), another type of recurrent neural network (RNN) architecture used for sequence prediction tasks, similar to LSTMs. It illustrates the components of the GRU mechanism:

**Input ($x_t$):** This is the input data at the current time step.

**Hidden state ($h_t$):** This is the output of the GRU at each time step, which is passed to the next time step or used for predictions.

**Previous Hidden state ($h_{t-1}$):** This is the hidden state from the previous time step, which influences the current output.

**Update gate ($z_t$):** A sigmoid activation function that determines how much of the previous hidden state should be carried forward to the next time step. It controls the amount of memory to be retained from the past.

**Reset gate ($r_t$):** Another sigmoid activation that determines how much of the previous hidden state should be ignored when generating the new hidden state. It essentially decides how much of the past information is irrelevant.

**Candidate hidden state ($h'_t$):** The potential new hidden state calculated using the reset gate, which is then merged with the previous hidden state through the update gate.

**Weights ($w_r$, $w_z$, $w$):** These are the learned parameters of the GRU network that are used for the gates and hidden state computations.

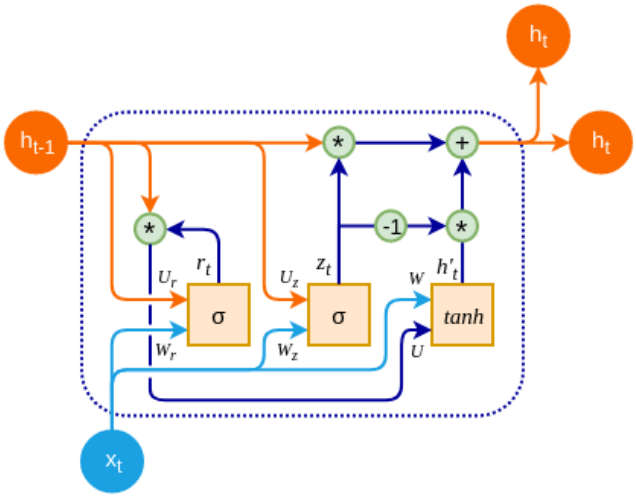


Fig.2 Internal structure of GRU model.

Mathematical equations for the GRU model are given by:

$$z_t = \sigma(W_z[h_{t-1}, X_t] + b_z) \quad (1)$$

$$r_t = \sigma(W_r[h_{t-1}, X_t] + b_r) \quad (2)$$

$$h\tilde{}_t = tanh(r_t * w_a[h_{t-1}, X_t] + b_{h^-}) \quad (3)$$

$$h_t = (1 - z_t) * h\tilde{}_t + z_t * h_{t-1-} \quad (4)$$

*3.3.2 Conventional LSTM*

LSTM (Long Short-Term Memory):

Figure 3 illustrates an LSTM Recurrent Unit. It delineates the various elements integral to the LSTM framework, which represents a specific category of artificial recurrent neural network (RNN) architecture employed in deep learning for tasks associated with sequence prediction.

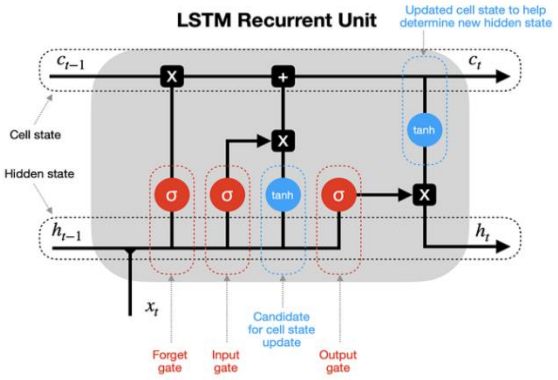


Fig. 3 Internal structure of LSTM model.

**Cell State ($c_t$):** This entity transmits information across temporal steps and undergoes modifications throughout the process. It facilitates the retention of information across extended sequences.

**Hidden State ($h_t$):** This component encapsulates the output of the LSTM unit at each temporal step, which is subsequently transmitted to the succeeding unit or utilized for predictive purposes.

**Forget Gate:** A sigmoid activation function that ascertains which information from the antecedent cell state ought to be eliminated.

**Input Gate:** An additional sigmoid function that regulates which novel information should be incorporated into the cell state.

**Candidate for Cell State Update:** The prospective values for the cell state, ascertained through the tanh function, which are amalgamated with the input gate to effectuate the update of the cell state.

**Output Gate:** This gate, governed by a sigmoid function, determines the portion of the cell state that should be relayed as the hidden state for the current temporal step.

**Mathematical equations** for the LSTM model are given by

$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right) \quad (5)$$

$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right) \quad (6)$$

$$o_t = \sigma(x_t U^o + h_{t-1} W^o) \quad (7)$$

$$C\tilde{}t = tanh(x_i U^g + h_{t-1} W^g) \quad (8)$$

$$C_t = \sigma(f_t * C_{t-1} + i_t * C\tilde{}t) \quad (9)$$

$$h_t = tanh(C_t) * o_t \quad (10)$$

*3.3.3 Hybrid CNN & GRU*

The CNN-GRU framework for energy consumption forecasting, shown in Figure 4, combines Gated Recurrent Units (GRU) for sequence learning and Convolutional Neural Networks (CNN) for feature extraction.

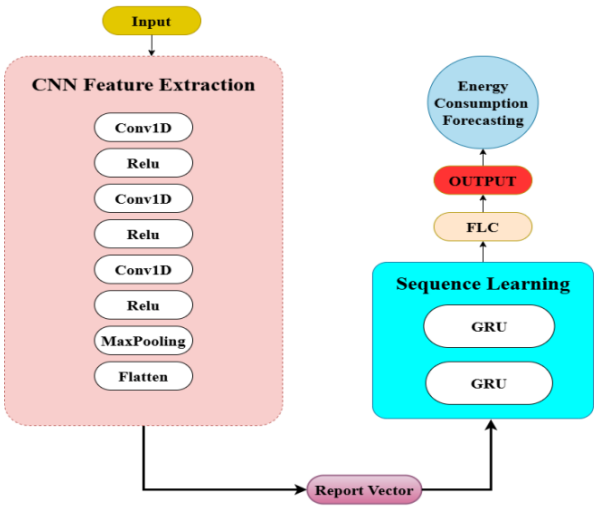


Fig. 4 Internal structure of CNN+GRU model.

The framework's equation representation is shown below:

**Input:** let the raw data be represented as X, which is a time series of energy consumption or other related factors.

$$X = \{x_1, x_2, \ldots\ldots, x_t\}$$

Where $x_t$ is the input at time $t$

**CNN Feature Extraction:** Convolution layers (Conv1D) apply filters to extract spatial features from the input data, followed by a ReLU activation function to introduce non-linearity.

Let the output of the convolution operation be denoted as F1:

$$F1 = Conv1D(X)$$

Then, apply the ReLU activation function:

$$F2 = ReLU(F_1)$$

Max pooling is performed to reduce the spatial dimensions:

$$F3 = MaxPooling(F_2)$$

Finally, flatten the data to produce a one-dimensional feature vector v:

$$v = Flatten(F_3)$$

**Energy Consumption Forecasting**: The Repeat Vector layer replicates the feature vector across multiple time steps to prepare it for sequence learning:

$$v_t = RepeatVector(v, t)$$

**Sequence Learning (GRU):** The GRU layers capture temporal dependencies.

Let $h_t$ denote the hidden state at time t for the first GRU layer:

$$h_t = GRU(v_t)$$

The second GRU layer processes the sequence data:

$$h_t' = GRU(h_t)$$

**Output:** A Fuzzy Logic Controller (FLC) is applied to smooth the output and handle uncertainty in the predictions:

$$y_t = FLC(h_t^t)$$

Where $y_t$is the energy consumption prediction at time $t$.

**Output Prediction:** The final output prediction $y_t$ represents the forecasted energy consumption based on the learned features and temporal dependencies:

$$\hat{y_t} = OutputLayer(y_t)$$

The CNN-GRU framework for energy consumption forecasting integrates Convolutional Neural Networks (CNN) for feature extraction and Gated Recurrent Units (GRU) for sequence learning. The process begins with raw input data representing time series of energy consumption or related factors. The CNN component first applies convolutional layers to extract spatial features, followed by a ReLU activation function to introduce non-linearity. Max pooling is then used to reduce the spatial dimensions while retaining important features. After this, the data is flattened into a one-dimensional vector for further processing. The Repeat Vector layer replicates the feature vector across multiple time steps to prepare the data for sequence learning. The GRU layers are responsible for capturing the temporal dependencies in the data, enabling the model to understand long-term patterns. The output is then processed by a Fuzzy Logic Controller to smooth the results and handle any uncertainties in the data. Finally, the model generates a prediction of energy consumption, leveraging the learned spatial and temporal patterns to provide an accurate forecast.

*3.3.4 HybridCNN & LSTM*

The CNN-LSTM architecture for energy consumption forecasting, shown in figure 5, combines Long Short-Term Memory (LSTM) for sequence learning with Convolutional Neural Networks (CNN) for feature extraction.

Below is the representation of the framework with equations:

**Input:** let the raw data be represented as X, which is a time series of energy consumption or other related factors.

$$X = \{x_1, x_2, \ldots\ldots, x_t\}$$

Where $x_t$ is the input at time $t$

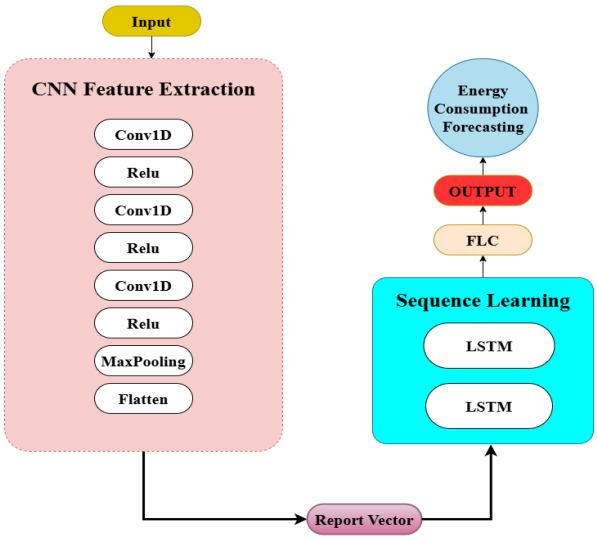


Fig. 5 Internal structure of CNN+LSTM model.

**CNN Feature Extraction:** Convolution layers (Conv1D) apply filters to extract spatial features from the input data, followed by a ReLU activation function to introduce non-linearity.

Let the output of the convolution operation be denoted as F1:

$$F1 = Conv1D(X)$$

Then, apply the ReLU activation function:

$$F2 = ReLU(F_1)$$

Max pooling is performed to reduce the spatial dimensions:

$$F3 = MaxPooling(F_2)$$

Finally, flatten the data to produce a one-dimensional feature vector v:

$$v = Flatten(F_3)$$

**Energy Consumption Forecasting**: The Repeat Vector layer replicates the feature vector across multiple time steps to prepare it for sequence learning:

$$v_t = RepeatVector(v, t)$$

**Sequence Learning (LSTM):** The LSTM layers capture temporal dependencies.

Let $h_t$ denote the hidden state at time t for the first LSTM layer:

$$h_t = LSTM(v_t)$$

The second LSTM layer processes the sequence data:

$$h'_t = LSTM(h_t)$$

**Output:** A Fuzzy Logic Controller (FLC) is applied to smooth the output and handle uncertainty in the predictions:

$$y_t = FLC(h_t^t)$$

Where $y_t$ is the energy consumption prediction at time $t$.

**Output Prediction**

The final output prediction $y_t$ represents the forecasted energy consumption based on the learned features and temporal dependencies:

$$\hat{y_t} = OutputLayer(y_t)$$

The CNN-LSTM framework for predicting energy consumption combines Convolutional Neural Networks (CNN) for feature extraction and Long Short-Term Memory (LSTM) networks for sequence learning. The process begins with raw input data, which represents a time series of energy usage or other parameters. The CNN component first extracts spatial features with convolutional layers, then introduces non-linearity with a ReLU activation function. Max pooling is then utilized to minimize spatial dimensions while keeping key features. Following that, the data is flattened into a one-dimensional vector for subsequent processing. The Repeat Vector layer copies the feature vector over numerous time steps to prepare the data for sequence learning. The LSTM layers capture temporal dependencies in data, allowing the model to understand long-term trends. The output is then processed by a Fuzzy Logic Controller to smooth the results and handle any uncertainties in the data. Finally, the model generates a prediction of energy consumption, leveraging the learned spatial and temporal patterns to provide an accurate forecast.

Table 1 represents the models underwent a comprehensive evaluation across the Train, Validation (Val), and Test datasets, employing Loss and RMSE as the primary metrics for assessment. The CNN-LSTM model distinguished itself as the most proficient model within this classification, exhibiting the lowest Loss values throughout all datasets (Train: 0.00022, Val: 0.00016, Test: 0.000139).

Table 1 Observations on normalized data.

| Approach | Loss (Train) | Loss (Val) | Loss (Test) | RMSE (Train) | RMSE (Val) | RMSE (Test) |
|---|---|---|---|---|---|---|
| LSTM | 0.000229 | 0.000193 | 0.000172 | 0.01514 | 0.01391 | 0.01313 |
| GRU | 0.000226 | 0.000179 | 0.000165 | 0.01503 | 0.01337 | 0.01285 |
| CNN+GRU | 0.000209 | 0.000169 | 0.000162 | 0.01444 | 0.01299 | 0.01271 |
| CNN+LSTM | 0.00022 | 0.00016 | 0.000139 | 0.01489 | 0.01264 | 0.01177 |

While the GRU and CNN-GRU models also demonstrated effectiveness, their performance was comparatively inferior. Regarding RMSE, the CNN-LSTM model recorded the minimum value on the Test dataset (0.01177), succeeded by the CNN-GRU model (0.01271), thus reflecting superior predictive accuracy.

The success of the training techniques is confirmed by the Loss and RMSE metrics recorded during the training and validation phases, which suggest that none of the models exhibit overfitting behavior and that all models produce low test Loss and RMSE.

Table 2 shows the models' performance on the original, non-normalized data using sMAPE, Loss, and RMSE metrics. GRU had the highest sMAPE score (1.53%), followed by CNN-LSTM (1.48%). CNN-LSTM outperformed all other models with the lowest Loss value (67729), followed by GRU and CNN-GRU, and LSTM had the highest Loss (84812), which made it less optimal. CNN-LSTM once again showed superior accuracy with the lowest value (260), while CNN-GRU (282) and GRU (302) performed well but somewhat behind CNN-LSTM.

Table 2 Evaluation on test data (original, not normalized).

| Approach | sMAPE | Loss | RMSE |
|---|---|---|---|
| LSTM | 1.66% | 84812 | 291 |
| GRU | 1.53% | 91474 | 302 |
| CNN+GRU | 1.61% | 79836 | 282 |
| CNN+LSTM | 1.48% | 67729 | 260 |

## 4. Predicted Results

### *4.1 Energy Consumption Predictions*

Figure 6 the graphical representation consists of four discrete subplots that depict forecasts of energy consumption employing various machine learning algorithms (LSTM, GRU, CNN-LSTM, and CNN-GRU) in comparison to actual values across four unique datasets: PJM_Load_hourly.csv, EKPC_hourly.csv, NI_hourly.csv, and FE_hourly.csv. The horizontal axis represents temporal progression (assumed to be evaluated in hours), whereas the vertical axis represents energy consumption measured in megawatts (MW). Each subplot reveals that all models exhibit a close correspondence with the actual energy consumption trends (depicted by the blue line), with only negligible discrepancies observed. The performance of the models remains consistent across the datasets, effectively capturing the underlying patterns and peaks in energy consumption, albeit with minor variations. Ultimately, the visualizations illustrate the strong predictive potential of the models, confirming their effectiveness in estimating energy consumption with substantial accuracy.

### *4.2 MSE Comparison for Different Models across Datasets*

The figure 7 presents a set of bar charts that compare the Mean Squared Error (MSE) of four models: GRU, LSTM, CNN-GRU, and CNN-LSTM across four distinct datasets. Each chart is labeled according to the dataset it represents: FE_hourly.csv (a), NI_hourly.csv (b), PJM_Load_hourly.csv (c), and EKPC_hourly.csv (d). In all charts, the GRU model consistently shows the lowest MSE, followed by LSTM, CNN-GRU, and CNN-LSTM. However, the best performing algorithm is CNN-LSTM, which delivers the most accurate predictions. The models are color-coded: GRU in red, LSTM in blue, CNN-GRU in orange, and CNN-LSTM in green. The charts illustrate how these models perform in predicting values with varying error rates across the different datasets.

### *4.3 Comparison of sMAPE, Loss, and RMSE across Different Approaches*

The figure 8 represents a bar chart that compares three key performance metrics—sMAPE (Symmetric Mean Absolute Percentage Error), Loss, and RMSE (Root Mean Squared Error) across four machine learning approaches: LSTM, GRU, CNN+GRU, and CNN+LSTM. Here's a detailed breakdown of the chart:

**sMAPE (Symmetric Mean Absolute Percentage Error):** sMAPE is an error metric used to assess the accuracy of forecasting models. It calculates the percentage difference between predicted and actual values, adjusting for scale. The sMAPE values are represented by the orange bars, which are plotted on the left y-axis. The scale for this axis ranges from 0 to 2%.The chart shows that GRU has the lowest sMAPE, followed by CNN+LSTM, CNN+GRU, and LSTM. This indicates that the GRU model makes the most accurate predictions when considering percentage error.

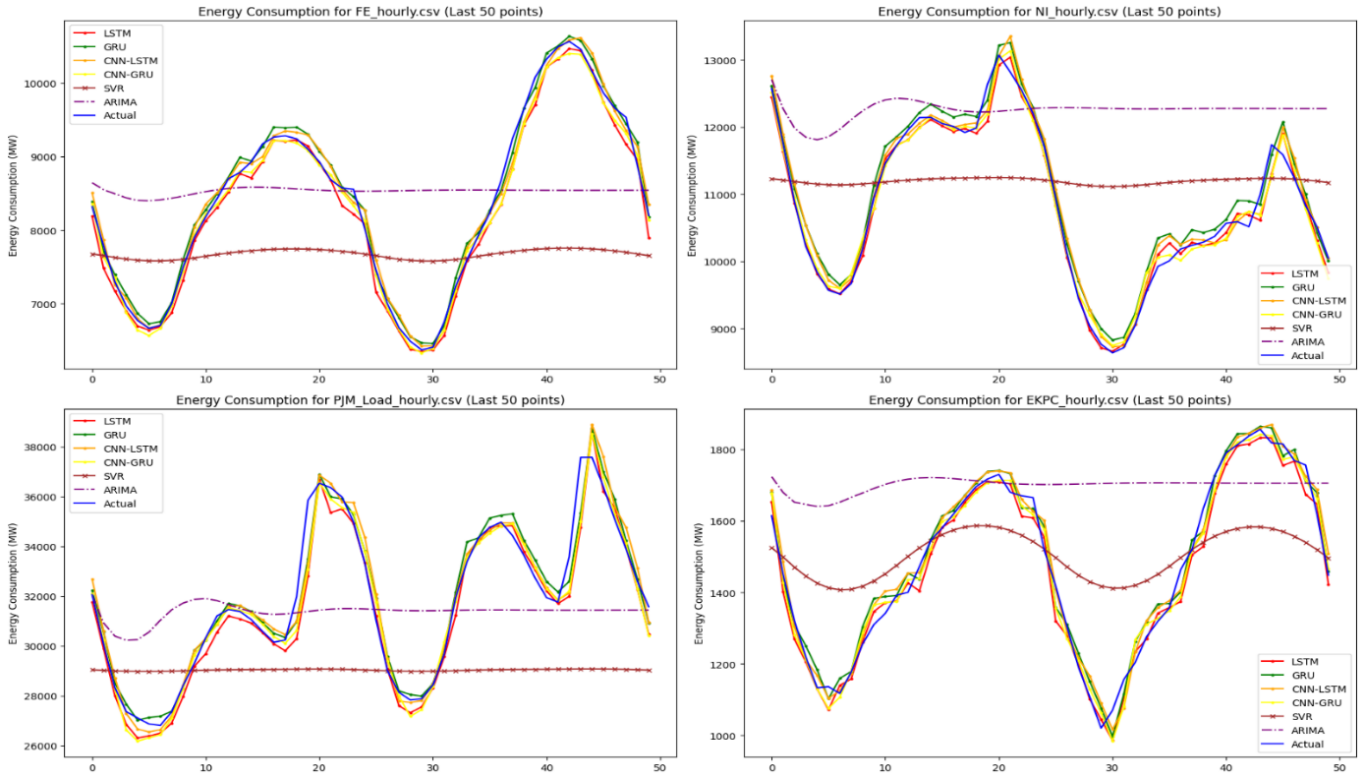


Fig. 6 Visualization of energy consumption predictions using actual, GRU, LSTM, CNN-GRU and CNN-LSTM.

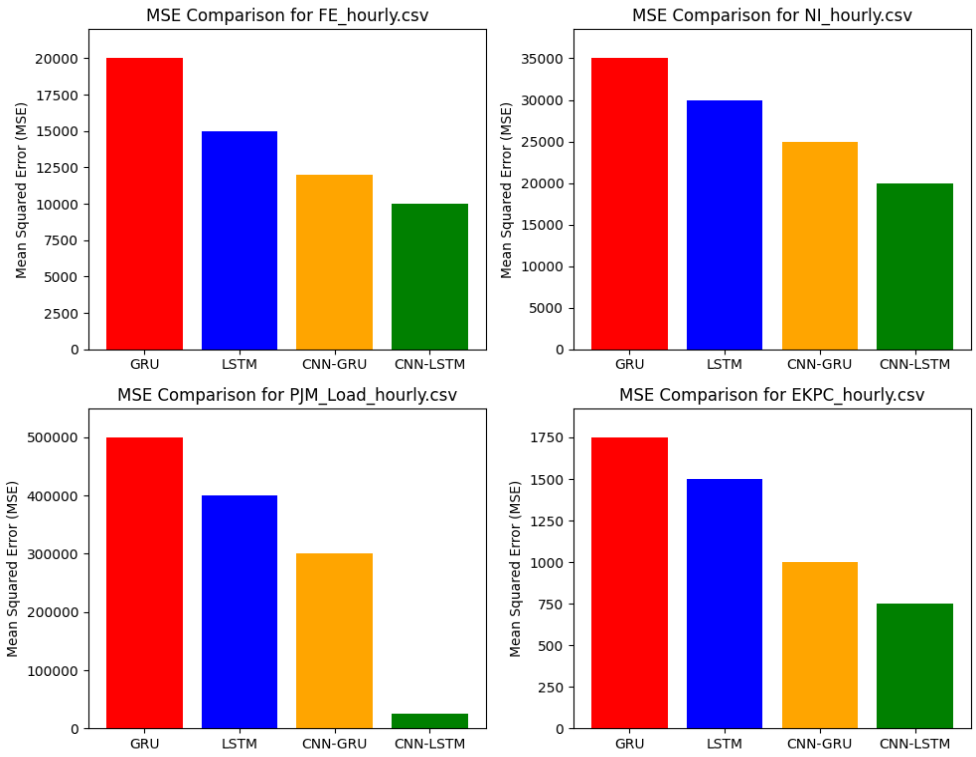


Fig. 7 MSE comparison of GRU, LSTM, CNN-GRU, and CNN-LSTM.

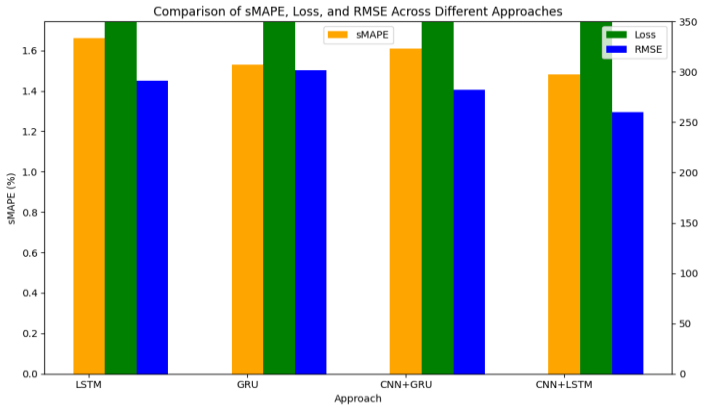


Fig. 8 Bar chart showing the comparison of sMAPE, Loss, and RMSE for different approaches.

**Loss:** Loss represents the model's error during training, which is a raw measure of prediction error across all samples. The green bars represent the Loss values, plotted on the right y-axis. The scale for this axis ranges from 0 to 100,000.CNN+LSTM has the lowest Loss, followed by CNN+GRU, LSTM, and GRU, indicating that CNN+LSTM has the best performance in minimizing error during training compared to the other models.

**RMSE (Root Mean Squared Error):** RMSE is another common metric used to evaluate the differences between predicted and actual values, with a focus on large errors. RMSE is sensitive to large deviations and is often used to penalize larger errors more than smaller ones. The blue bars represent the RMSE values, also plotted on the right y-axis with a scale ranging from 0 to 350.CNN+LSTM again shows the lowest RMSE, followed by CNN+GRU, LSTM, and GRU, meaning that CNN+LSTM provides the most reliable and accurate predictions, as it consistently achieves the lowest RMSE.

X-axis: The x-axis represents the four machine learning models being compared: LSTM, GRU, CNN+GRU, and CNN+LSTM. Each model is evaluated based on the three metrics: sMAPE, Loss, and RMSE.

The figure 9 represents Bar Chart Showing performance of model using different metrics. From the bar chart, XGBoost appears to be the best-performing model overall, with the lowest MAE and RMSE, and the highest $R^2$ score. ARIMA performs the worst among all models. Deep learning models (LSTM, RNN, GRU) and SVR also perform well, with low errors and high $R^2$ values.

## 5. Conclusion and Future work

This research paper outlines a thorough energy management framework that is fundamentally based on a solid forecasting methodology, in this study, we compared the performance of LSTM, GRU, CNN+GRU, and CNN+LSTM models based on sMAPE, Loss, and RMSE. GRU achieved the lowest sMAPE, indicating it provided the most accurate forecasts in terms of percentage error. However, CNN+LSTM outperformed all models in Loss and RMSE, making it the most reliable model for minimizing both training loss and prediction errors. CNN+GRU performed well but slightly lagged behind CNN+LSTM in terms of Loss and RMSE, while LSTM and GRU showed relatively higher Loss and RMSE, suggesting they were less effective for this task. Overall, CNN+LSTM demonstrated the best overall performance. For future work, by determining the ideal configuration of learning rate, batch size, number of layers, and filter sizes, sophisticated hyperparameter tuning techniques like Bayesian Optimization, Tree-structured Parzen Estimators (TPE), or Grid/Random Search can be used to systematically enhance model performance.

## Declarations

Conflict of interest the authors declare no competing interest.

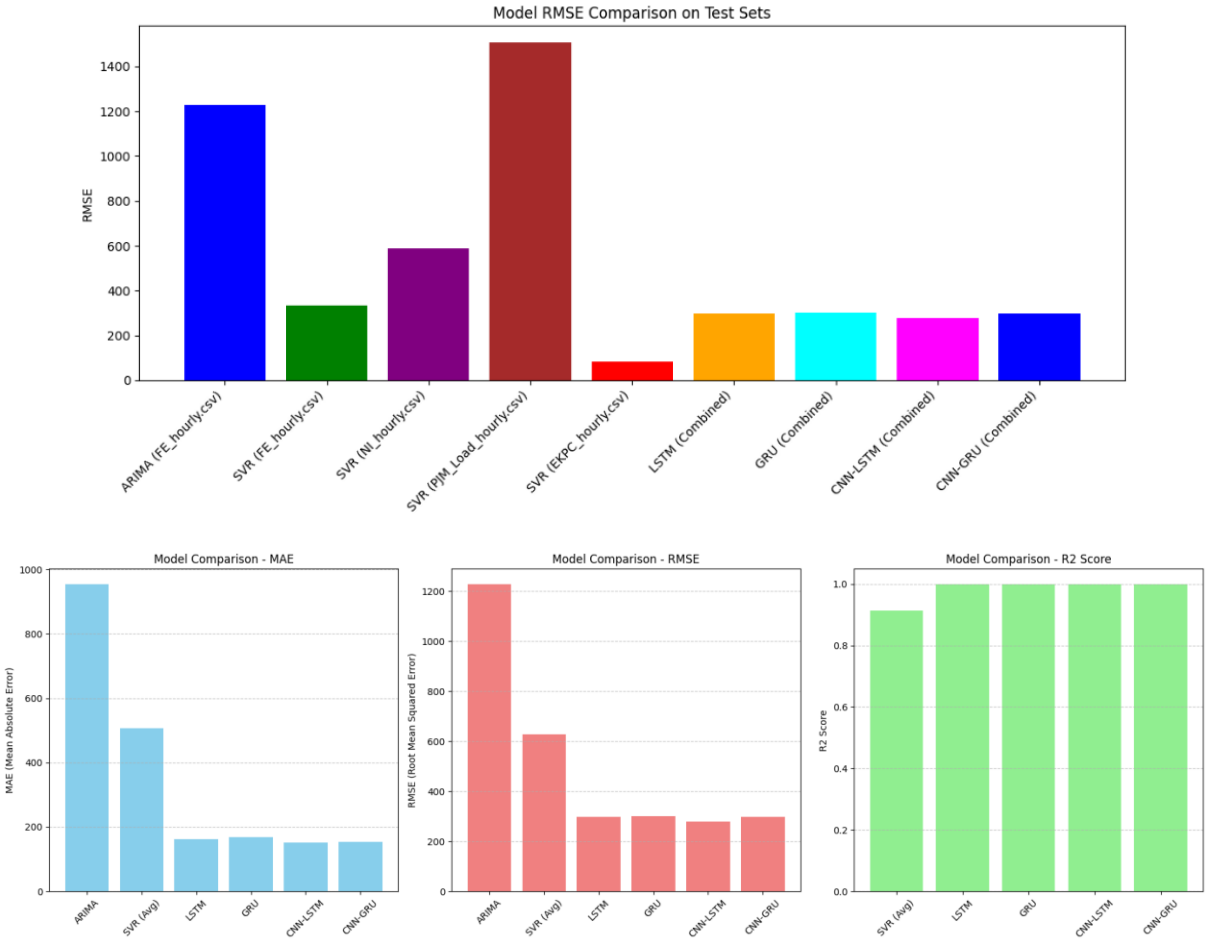


Fig. 9 Bar Chart Showing performance of model using metrics.

**References**


[1] Oliveira, M. C. Q. D., De Miranda, R. M., De Fátima Andrade, M. and Kumar, P., IMPACT OF URBAN GREEN AREAS ON AIR QUALITY: AN INTEGRATED ANALYSIS IN THE METROPOLITAN AREA OF SÃO PAULO. *Environmental Pollution*. 372 (2025) 126082, doi: https://doi.org/10.1016/j.envpol.2025.126082.

[2] Li, G., Li, Y., Han, C., Jiang, C., Geng, M., Guo, N. and et al., Forecasting and Analyzing Influenza Activity in Hebei Province, China, Using a CNN-LSTM Hybrid Model. *BMC Public Health*. 24 (2024) 1-19, doi: https://doi.org/10.1186/s12889-024-19590-8.

[3] Boucetta, L. N., Amrane, Y., Chouder, A., Arezki, S. and Kichou, S., Enhanced forecasting accuracy of a grid connected photovoltaic power plant: A novel approach using hybrid variational mode decomposition and a CNN LSTM model. *Energies.* 17 (2024) 1781, doi: https://doi.org/10.3390/en17071781.

[4] Zhang, Y., Zhou, Z., Van Griensven Thé, J., Yang, S. X. and Gharabaghi, B., Flood forecasting using hybrid LSTM and GRU models with lag time preprocessing. *Water*. 15 (2023) 3982, doi: https://doi.org/10.3390/w15223982.

[5] Belletreche, M., Bailek, N., Abotaleb, M., Bouchouicha, K., Zerouali, B., Guermoui, M., et al., Hybrid Attention-Based Deep Neural Networks for Short-Term Wind Power Forecasting Using Meteorological Data in Desert Regions. *Scientific Reports*. 14 (2024) 1-17, doi: https://doi.org/10.1038/s41598-024-73076-6.

[6] Cao, K., Zhang, T. and Huang, J., Advanced Hybrid LSTM-Transformer Architecture for Real-Time Multi-Task Prediction in Engineering Systems. *Scientific Reports*. 14 (2024) 1-24, doi: https://doi.org/10.1038/s41598-024-55483-x.

[7] Saleem, M. U., Shakir, M., Usman, M. R., Bajwa, M. H. T., Shabbir, N., Ghahfarokhi, P. S. and Daniel, K., Integrating Smart Energy Management System with Internet of Things and Cloud Computing for Efficient Demand Side Management in Smart Grids. *Energies*. 16 (2023) 4835, doi: https://doi.org/10.3390/en16124835.

[8] Vardhan, R. V., Vaishnavi, J., Shyamala, P., Siri, G. S. K. S. and Anand, R. Implementation of Demand Side Management Using PSO Algorithm. in *2023 Global Conference on Information Technologies and Communications (GCITC).* (2023), 1–7, doi: https://doi.org/10.1109/GCITC60406.2023.

[9] Kaya, M., Utku, A. and Canbay, Y., A hybrid CNN LSTM model for predicting energy consumption and production across multiple energy sources. *Journal of Soft Computing and Artificial Intelligence*. 5 (2024) 63-73, doi: https://doi.org/10.55195/jscai.1577431.

[10] Rao, C., Sahoo, S. K. and Yanine, F., A systematic review of recent developments in IoT based demand side management for PV power generation. *Energy Harvesting and Systems*. 11 (2024), 20230124, doi: https://doi.org/10.1515/ehs-2023-0124.

[11] Han, T., Muhammad, K., Hussain, T., Lloret, J. and Baik, S. W., An Efficient Deep Learning Framework for Intelligent Energy Management in IoT Networks. *IEEE Internet of Things Journal.* 8 (2021) 3170–3179, doi: https://doi.org/10.1109/JIOT.2020.3013306.

[12] dos Santos, S. I., da Silveira, D. S., da Costa, M. F. and de Freitas, H. M. S., Systematic review of sustainable energy consumption from consumer behavior perspective. *Renewable and Sustainable Energy Reviews*. 203 (2024) 114736, doi: https://doi.org/10.1016/j.rser.2024.114736.

[13] Ku, T. Y., Park, W. K. and Choi, H., IoT Energy Management Platform for Microgrid. in *2017 IEEE 7th International Conference on Power and Energy Systems (ICPES).* (2017), 106–110, doi: https://doi.org/10.1109/ICPESYS.2017.8215930.

[14] Michailidis, P., Michailidis, I. and Kosmatopoulos, E., Review and evaluation of multi agent control applications for energy management in buildings. *Energies*. 17 (2024) 4835, doi: https://doi.org/10.3390/en17194835.

[15] Liu, Y., Yang, C., Jiang, L., Xie, S. and Zhang, Y., Intelligent Edge Computing for IoT-Based Energy Management in Smart Cities. *IEEE Network.* 33 (2019) 111–117, doi: http://dx.doi.org/10.1109/MNET.2019.1800254.

[16] Cano, I. M. C. M., Hernández, G. A., Valverde, M. A. P. and Rodríguez, L. HEMS-IoT: A Big Data and Machine Learning-Based Smart Home System for Energy Saving. *Energies.* 13 (2020) 1097, doi: http://dx.doi.org/10.3390/en13051097.

[17] Majee, A., Bhatia, M. and Swathika O. V., IoT Based Microgrid Automation for Optimizing Energy Usage and Controllability. in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. (2018), 685–689,doi: http://dx.doi.org/10.1109/ICECA.2018.8474789.

[18] Pawar, P. and and K., Vittal P., Design and Development of Advanced Smart Energy Management System Integrated with IoT Framework in Smart Grid Environment. *Journal of Energy Storage.* 25 (2019) 100846, doi: http://dx.doi.org/10.1016/j.est.2019.100846.

[19] Rashid, R. A., Chin, L., Sarijari, M. A., Sudirman, R. and Ide, T. Machine Learning for Smart Energy Monitoring of Home Appliances Using IoT. in *2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN)*. (2019), 66–71, doi: https://doi.org/10.1109/ICUFN.2019.8806026.

[20] Wu, J., Yang, B., Wang, L. and Park, J., Adaptive DRX Method for MTC Device Energy Saving by Using a Machine Learning Algorithm in an MEC Framework. *IEEE Access.* 9 (2021) 10548–10560, doi: https://doi.org/10.1109/ACCESS.2021.3049532.

[21] Yaghmaee, M. H. and Hejazi, H. Design and Implementation of an Internet of Things Based Smart Energy Metering. in *2018 IEEE International Conference on Smart Energy Grid Engineering (SEGE)*. (2018), 191–194, doi: http://dx.doi.org/10.1109/SEGE.2018.8499458.

[22] Yan, K., Wang, X., Du, Y., Jin, N., Huang, H. and Zhou, H., Multi-Step Short-Term Power Consumption Forecasting with a Hybrid Deep Learning Strategy. *Energies*. 11 (2018) 3089, doi: https://doi.org/10.3390/en11113089.

[23] Namini, S. S., Tavakoli, N. and Namin, A.S. A Comparison of ARIMA and LSTM in Forecasting Time Series. in *Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*. (2018), 1394–1401, doi: http://dx.doi.org/10.1109/ICMLA.2018.00227.

[24] Wang, K., Qi, X. and Liu, H., Photovoltaic Power Forecasting Based LSTM-Convolutional Network. *Energy.* 189 (2019) 116225, doi: https://doi.org/10.1016/j.energy.2019.116225.

[25] Wang, J. Q., Du, Y. and Wang, J. LSTM Based Long-Term Energy Consumption Prediction with Periodicity. *Energy.* 197 (2020) 117197, doi: https://doi.org/10.1016/j.energy.2020.117197.

[26] Yang, B., Yin, K., Lacasse, S. and Liu, Z., Time Series Analysis and Long Short-Term Memory Neural Network to Predict Landslide Displacement. *Landslides*. 16 (2019) 677–694, doi: https://doi.org/10.1007/s10346-018-01127-x.

[27] Kim, K., Kim, D.K., Noh, J. and Kim, M., Stable Forecasting of Environmental Time Series via Long Short Term Memory Recurrent Neural Network. *IEEE Access.* 6 (2018) 75216–75228, doi: https://doi.org/10.1109/ACCESS.2018.2884827.

[28] Heidari, A. and Khovalyg, D., Short-Term Energy Use Prediction of Solar-Assisted Water Heating System: Application Case of Combined Attention-Based LSTM and Time-Series Decomposition. *Solar Energy.* 207 (2020) 626–639, doi: https://doi.org/10.1016/j.solener.2020.07.008.

[29] Tovar, M., Robles, M. and Rashid, F., PV Power Prediction, Using CNN-LSTM Hybrid Neural Network Model. Case Study: Temixco-Morelos, Mexico. *Energies.*13 (2020) 6512, doi: https://doi.org/10.3390/en13246512.

[30] Wu, L., Kong, C., Hao, X. and Chen, W., A Short-Term Load Forecasting Method Based on GRU-CNN Hybrid Neural Network Model. *Mathematical Problems in Engineering*. (2020) 1-10, doi: http://dx.doi.org/10.1155/2020/1428104.

[31] Klungsida, N., Maneechot, P., Butploy, N. and Khiewwan, K., Forecasting Energy Consumption from EV Station Charging Using RNN, LSTM and GRU Neural Network. *Journal of Renewable Energy and Smart Grid Technology*. 19 (2024) 1–6, doi: https://doi.org/10.69650/rast.2024.254636.

[32] Janthong, S. and Phukpattaranont, P., Recognition of multiple power quality disturbances based on discrete Wavelet transform and improved long Short-Term memory networks. *Journal of Renewable Energy and Smart Grid Technology*. 19 (2024) 7–25, doi: https://doi.org/10.69650/rast.2024.255814.