Day-Ahead Electricity Demand Forecasting Using LSTM Deep Learning Model

Thi Lan Hương Pham[1,3], Viet Thien An Nguyen[2], Ba Hau Vu[1], and Dang Thanh Bui[1,2,*]

ARTICLE INFO

Article history:
Received: 29 March 2024
Revised: 8 July 2024
Accepted: 23 July 2024
Online: 30 June 2025

Keywords:
Load forecasting
LSTM
RNN
Energy demand forecast

ABSTRACT

This study compares the performance of recurrent neural network (RNN) models such as Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and their bidirectional variants (Bi-LSTM and Bi-GRU) in comparison to feedforward neural networks (FFNNs) for predicting load demand. FFNNs cannot capture temporal dependencies, whereas RNNs can effectively model sequential data. The results show that RNN-based models outperform FFNNs in load demand forecasting, handling non-linear and dynamic patterns more effectively. The advantage of RNNs lies in their ability to store past data to forecast future values, enabling them to understand complex sequential patterns and provide accurate predictions. Therefore, RNNs offer flexibility to FFNNs in forecasting load demand for modern power systems. With a root mean square error of 59.745 kW, a mean absolute error of 42.652 kW, and a mean absolute percentage error of 6.374%, this study providing valuable insights into deep learning techniques for load demand forecasting, which is essential for power companies and grid operators to cope with the complexities of modern energy systems.

1. INTRODUCTION

Accurate load forecasting is crucial for utilities and grid operators. It allows for optimized resource allocation, cost reduction, and streamlined utility operations. Grid operators rely on accurate load forecasts to maintain grid stability, prevent overloads, and avoid blackouts [1]. Additionally, accurate load forecasting helps utilities optimize cleaner energy sources, reducing environmental impact. Overall, accurate load forecasting is essential for creating sustainable, cost-effective, and resilient energy systems that balance supply and demand and meet the evolving needs of societies.

Vietnam's average daily electricity usage in August 2023 was estimated at around 825.8 million kWh, representing a 7.3% rise relative to the corresponding period in 2022 [2]. North Vietnam experienced an electrical shortfall in the summer because of increased demand, a lack of backup supplies, and the effects of climate change. According to predictions, this region may experience a shortage of 2,000 MW in the next two years [3]. Therefore, predicting electricity consumption in Vietnam is crucial and necessary to stabilize and regulate the power grid, ensuring a reliable supply for industrial production and daily activities. Due to the lack of reliable load data, underdeveloped infrastructure, and limited attention to load monitoring in Vietnam, we selected a reliable dataset from a building in Tsukuba, Japan, which shares similarities with Vietnam regarding cultural and temporal patterns [4].

Accurate electricity demand forecasting is now achievable with advanced technologies like recurrent neural network (RNN) models. Several studies have been conducted in this field to enhance forecasting accuracy. One such study, referenced as [5], proposes a methodology that utilizes neural networks and multi-criteria analysis to forecast electricity demand. The research compares different forecasting models and incorporates socioeconomic variables as inputs, with the Extreme Learning Machine (ELM) network demonstrating superior performance. An additional study [6] highlights the critical role of precise load forecasting in ensuring effective power system planning and operational efficiency. The authors investigate the use of RNN algorithms for electrical load forecasting, training and testing the models with accurate data. The findings indicate that the Gate Recurrent Unit (GRU) model, which uses two hidden layers and a learning rate of 0.01, performs better than the others, showing high R-squared values and low error metrics. Research [7] introduces a sequence-to-sequence (S2S) approach that applies deep learning methods for forecasting building energy consumption. GRU and Long Short-Term Memory-based S2S models are compared to traditional RNN and Neural Network (NN) models, with the GRU and LSTM-based S2S models exhibiting superior accuracy. Another study [8]

[1]*Institute for Control Engineering and Automation, Hanoi University of Science and Technology, Hanoi, 11615, Vietnam.*
[2]*School of Electrical and Electronic Engineering, Hanoi University of Science and Technology, Hanoi, 11615, Vietnam.*
[3]*Vietnam National University of Agriculture, Hanoi, 12406, Vietnam.*
**Corresponding author:* Dang Thanh Bui; Phone: +84-915-897-699; Email: thanh.buidang@hust.edu.vn.

investigates energy consumption prediction by utilizing a range of neural network models, including LSTM, multi-layer GRU, and Drop-GRU. The authors compare these models' accuracy and computational efficiency, finding that the Drop-GRU model outperforms the LSTM and GRU models regarding accuracy and prediction speed. Deep learning models, specifically RNN and LSTM networks, are discussed in [9] for electricity consumption prediction. The study compares these models with popular prediction models and finds that the RNN and LSTM models achieve an average RMSE of 0.1 kW. The models are tested on individual houses and blocks of houses for short, mid, and long-term predictions. Moreover, LSTM models can learn from historical data to identify patterns and trends unique to a specific building or region, allowing for more accurate and personalized forecasts [10]. For instance, an LSTM model can learn to recognize the daily and weekly energy consumption patterns in a building and the seasonal variations in energy demand that occur during different times of the year [11].

Furthermore, LSTM models can be trained with multiple inputs to give a better understanding of the factors affecting electricity demand [12]. Finally, research [13] presents a neural basis expansion analysis for forecasting short-term energy consumption in grids. The proposed RNN-based models improve prediction accuracy by incorporating covariates and handling large datasets. It surpasses other neural network-based methods and demonstrates superior accuracy in daily, weekly, and monthly energy consumption predictions.

In this research, we use FFNN and RNN_based models to forecast a building's electricity demand. The remaining section is organized as follows: Section 2 illustrates the methodology; Section 3 explains the data preparation, experiments, and corresponding results; and Section 4 concludes the paper.

2. METHODOLOGY

2.1. System overview

In this context, the suggested deep learning model for predicting energy demand for the next day uses a matrix shown in Fig. 1, with dimensions of 24 by n. The variable 'n' denotes the number of pertinent features employed for prediction, capturing the intricate interdependence of factors directly influencing energy demand. These features encompass energy elements such as humidity, temperature, pressure, wind speed, solar irradiation, and past energy values. Additionally, temporal aspects like hour of the day (HOD) or month of the year (MOY), which significantly impact forecasting outcomes, are integrated into the model. The model generates outputs as a vector with a length of 24, representing 24 future values at 1-hour intervals.

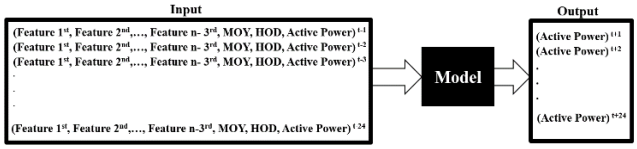


Fig. 1. Deep learning model structure for multivariate data.

2.2 Feed-Forward Neural Network

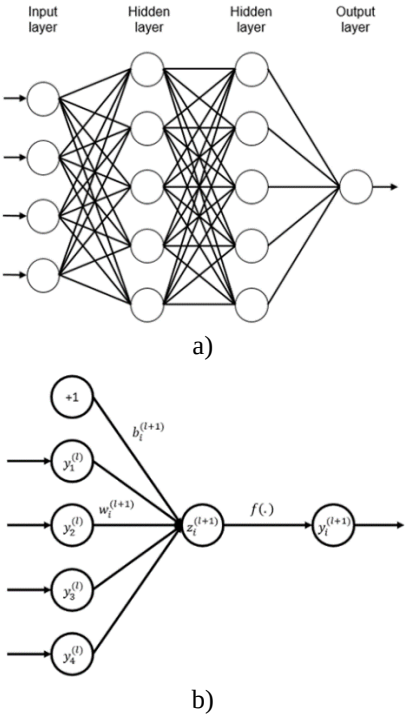


Fig. 2. FFNN structure (a) and FFNN process (b).

When multiple hidden layers are added, FFNNs are commonly referred to as multilayer perceptrons (MLPs) or deep feed-forward neural networks (FFNNs). Fig. 2a illustrates the general structure of the FFNN model. When each layer contains many neurons (units), there are three types: input, output, and hidden. Every connection between neurons is assigned to weight and a bias.

Take into account an FFNN with L hidden layers, $l \in \{1,...,L\}$ is the index of the hidden layers. The forward propagation process of FFNN is illustrated in Fig. 2b and Equation (1) (2):

$$z_i^{(l+1)} = w_i^{(l+1)} y^{(l)} + b_i^{(l+1)} \quad (1)$$

$$y_i^{(l+1)} = f(z_i^{(l+1)}) \quad (2)$$

where, $z^{(l)}$: the vector of inputs, $y^{(l)}$: the vector of outputs, i : any hidden unit, $W^{(l)}$: weight matrix of layer l , $b^{(l)}$: a bias vector of layer l, $f(.)$: activation function.

In the backpropagation stage, a gradient descent method trains the neural network. The primary objective of this step

is to minimize the error function $J(W,b)$ such as cross-entropy for classification tasks and mean-squared error for regression tasks. The gradient descent algorithm adjusts the parameters in each iteration, as represented in Equation (3) [14].

$$W_{ij}^{(l)} \leftarrow W_{ij}^{(l)} - \alpha \frac{\partial}{\partial w_{ij}^{(l)}} J(W,b), b_{ij}^{(l)} \leftarrow b_{ij}^{(l)} - \alpha \frac{\partial}{\partial b_{ij}^{(l)}} J(W,b). \quad (3)$$

where, $W_{ij}^{(l)}$: an element of the weight matrix associated with the connection between unit j in layer l and unit i in layer $l+1$, $b_i^{(l)}$: an element of the bias vector associated with the unit i in layer $l+1$ and α is the learn rate.

FFNN models are also widely used in time-series data forecasting. In this context, this network is commonly used as a predictive tool for time-series forecasting, where it utilizes past observations to estimate future values. The observed values are used as inputs to the neural network, and through hidden layers, the network learns to create hidden models that capture complex relationships between the observed values and the forecasted future values. During training, FFNN fine-tunes the weights and biases of its neural connections to enhance the accuracy of time-series predictions. Techniques such as backpropagation are used to optimize the network's parameters throughout this process.

2.3 Recurrent Neural Network

RNN models have been used for modeling sequential data in Deep Learning. They were commonly recommended before attention models emerged. RNNs have the advantage of weight sharing across sequence elements, allowing them to handle sequences of varying lengths and generalize effectively. Due to their versatile architecture, RNNs can also be applied to structured data types like geographical or graphical data. This versatility makes RNNs a popular choice for modeling various types of data.

Those are a class of neural networks that incorporate hidden states, enabling them to use previous outputs as inputs for the current step. Their typical structure is illustrated in Fig. 3.

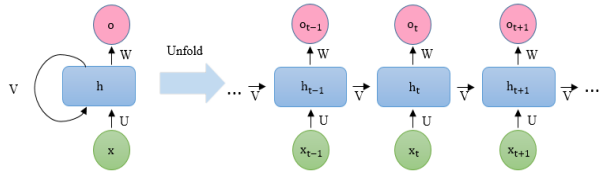


Fig. 3. RNN structure.

At each timestep t, the activation $a^{\langle t \rangle}$ and the outputs $y^{\langle t \rangle}$ are expressed as Equation (4) and Fig. 4 [15]:

$$a^{\langle t \rangle} = g_1(W_{aa}a^{\langle t-1 \rangle} + W_{ax}x^{\langle t \rangle} + b_a), y^{\langle t \rangle} = g_2(W_{aa}a^{\langle t \rangle} + b_y). \quad (4)$$

where, W_{ax}, W_{aa}, W_{ya}, b_a, b_y are coefficients that are shared temporally and g_1, g_2 are activation functions.

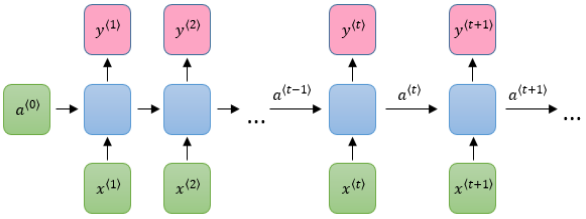


Fig. 4. RNN process.

The RNN framework, while effective for modeling sequential data, faces particular challenges that have prompted the development of variations aimed at addressing these limitations. Traditional RNNs face two major challenges: struggling to retain information over long sequences and dealing with vanishing or exploding gradients. These gradient issues arise from the backpropagation process used during training. As the number of time steps increases, the gradients moving through the network either shrink or grow exponentially, which makes it difficult for the network to effectively learn and recognize patterns over long sequences. This limitation restricts the ability of RNNs to model long-term dependencies accurately. To mitigate the challenges associated with the vanishing and exploding gradient problems, as well as to effectively capture long-term dependencies in traditional RNNs, researchers have introduced several modifications, particularly the LSTM and GRU architectures. These variants introduce specialized gating mechanisms to update and retain information over multiple time steps selectively.

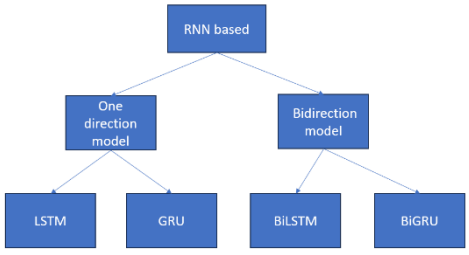


Fig. 5. Variations of RNN.

Fig. 5 illustrates the architecture of these variations and their flow of information, showcasing the incorporation of memory cells, gating units, and bidirectional processing. These modifications have been proven effective in mitigating the challenges faced by traditional RNNs, enabling more robust modeling of sequential data, particularly in time-series prediction tasks. Several variations of RNNs have been updated and maintained in an internal state and were developed to address the issue of gradient vanishing.

2.3.1 LSTM model

The LSTM cell is a specialized form of RNN designed to capture and model long-term dependencies within sequential data. It was first introduced by Hochreiter and Schmidhuber in 1997 [16]. The core idea of LSTM is to enable the network to learn how to retain crucial information in its long-term state, eliminate irrelevant data, and extract meaningful insights from it.

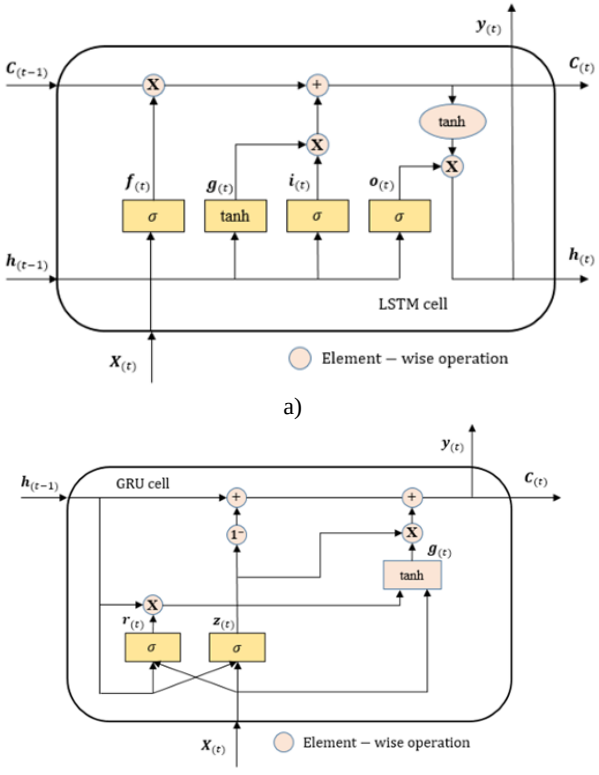


Fig. 6. The architecture of an LSTM cell (a) and gated recurrent unit (GRU) cell (b).

The architectural configuration of an LSTM cell is depicted in Fig. 6a. An LSTM cell comprises four key components, namely the input gate *i(t)*, forget gate *f(t)*, cell candidate *i(t)*, and output gate *o(t)*. The input gate determines which input portions should be incorporated into the cell's memory. The forget gate manages whether information from the previous cell state should be retained or discarded. The cell candidate generates potential new information to be incorporated into the cell state. Lastly, the output gate decides what proportion of the cell state should be passed on to the next step or used in the final prediction.

This architectural design of the LSTM cell enables it to effectively capture and model long-term dependencies, making it a valuable tool for various tasks involving sequential data analysis and prediction [17].

2.3.2 GRU model

The architecture of the GRU cell was introduced by Kyunghyun Cho et al. [18]. The GRU cell can be viewed as a simplified variant of the LSTM cell while maintaining comparable performance. Fig. 6b illustrates the structure of a GRU cell. In contrast to the LSTM cell, the GRU cell does not possess an explicit output gate. Instead, it combines long-term and short-term states into a single cell state. The GRU cell includes three primary elements: the reset gate, the update gate, and the cell candidate. The reset gate determines the extent to which the previous hidden state should be reset or discarded, helping the cell forget unnecessary information. The update gate regulates how much of the previous and current hidden states should influence the cell state, balancing the integration of new information. Lastly, the cell candidate produces potential new information to be incorporated into the cell state.

2.3.3 BiRNN-based model

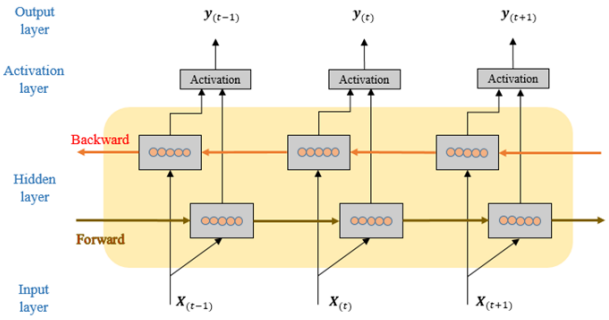


Fig. 7. A fundamental architecture of a basic BiRNN.

The preceding sections have elucidated the concept of unidirectional RNNs, wherein information flows directly from previous states to subsequent states. However, a prevalent requirement in numerous practical applications, particularly in sequence-to-sequence learning tasks, is the ability to generate predictions that rely on the entire input sequence. To address this necessity, bidirectional RNNs (BiRNNs) were proposed [19]. A fundamental architectural representation of a basic BiRNN is depicted in Fig. 7. Essentially, the network integrates two independent RNNs. The first RNN progresses forward in time, while the second RNN operates backward through time.

Inspired by the concept of BiRNNs, Bidirectional LSTM (BiLSTM) and Bidirectional GRU (BiGRU) were developed as extended versions of LSTM and GRU networks. In these models, two distinct layers are used: one processes the input sequence from left to right, while the other processes it from right to left. The results from both layers are then combined to form the final output, allowing the model to capture contextual information from both past and future time steps.

The bidirectional architecture of LSTM and GRU models renders them particularly effective for load demand forecasting tasks. By capturing temporal dependencies and daily/weekly cycles, these models can incorporate

contextual information from past and future time steps, providing a more comprehensive understanding of load demand patterns. Furthermore, their ability to handle non-linear relationships between variables enables them to forecast load demand accurately, even for complex and dynamic factors. Taking into account both forward and backward dependencies in these models leads to more precise predictions, while their resilience to noise and outliers ensures consistent and reliable performance.

2.4 Evaluation metrics

The correlation of data and its processing into a model within the ML pipeline enables the automation of the machine learning workflow, resulting in the generation of outputs. The MAE, RMSE, and MAPE metrics are utilized to evaluate RMSE values, which indicate superior model performance. RMSE offers immediate performance details for prediction models and is preferred to have a value close to zero, always remaining positive [20].

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - x_i\right)^2} \quad (5)$$

where, y_i is predicted value, x_i is measured value and n is the number of observations.

MAE gauges the average size of errors between paired values that correspond to the same event or instance. This metric helps assess how much the predicted values deviate, on average, from the actual values. To compute MAE, the absolute differences between each predicted value and its actual counterpart are calculated and then averaged. By doing this, it ensures that all errors are treated as positive, reflecting the overall scale of the discrepancies without accounting for whether they are overestimations or underestimations.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - x_i\right| \quad (6)$$

MAPE is a popular metric for assessing the accuracy of predictions in forecasting systems. It expresses accuracy as a percentage, providing a clear measure of how close predictions are to actual values. To calculate MAPE, the absolute difference between the predicted and actual values is first determined, then divided by the actual values, and the average of these percentage errors is computed across all time periods.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\frac{\left|y_i - x_i\right|}{y_i} \quad (7)$$

3. EXPERIMENT AND RESULTS

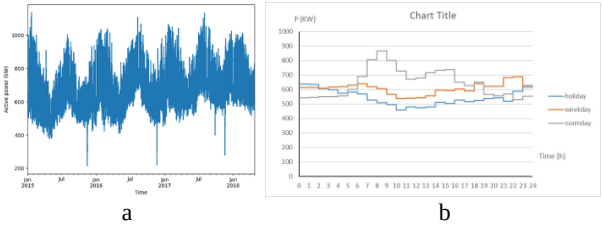
3.1. Data preparation

As previously mentioned, the dataset utilized in this study is derived from a building located in Tsukuba, Japan [4]. The dataset used in this study includes information on electricity usage, battery operations (charging and discharging), solar energy production, solar irradiation on an hourly basis, and electricity pricing. The data has been resampled to 1-hour resolution for predicting power consumption over the next seven days. The dataset contains 29,040 measurements of power consumption. Weather features and solar irradiation are also provided in Table 1 [21]. These variables are crucial in shaping energy consumption patterns across different climates and are essential for energy planning and resource management.

Table 1. Data features.

Data	Units
Ambient temperature	deg C
Humidity	%
Speed of wind	m/s
Solar irradiation	W/m^2
Active power	kW

Fig. 8a represents the power consumed data collected from January 2015 to April 2018 with a one-hour timestep resolution. Fig. 8b compares power consumption across three distinct days: holidays, weekends, and regular days. The data indicates that on normal days, the building exhibits the highest energy consumption during working hours, spanning from 7 a.m. to 6 p.m. Conversely, both holidays and weekends demonstrate a lower level of energy consumption.



a b

Fig. 8. The active power of building from 2015 to April 2018 (a) and three kinds of days in 2017 (b).

3.1.1 Data preprocessing

Comprehensive data preprocessing is essential for enhancing the performance of models in time series forecasting. It involves organizing the data chronologically, handling missing values, and normalizing or standardizing it to ensure consistent scales across features. Temporal patterns are captured by constructing sequences of input data representing temporal windows. The dataset is subsequently split into training and testing sets for the purpose of model evaluation. Additional preprocessing steps, such as handling categorical variables and encoding, may be needed depending on the data. Proper preprocessing enables the model to effectively learn the complex temporal

dependencies within the time series, enhancing its forecasting capabilities.

3.1.2 Data splitting

It is essential to partition the data into distinct training, validation, and test sets to ensure the effectiveness of time series forecasting. The training set is employed to develop forecasting models using historical data, while the validation set serves to periodically evaluate and refine the model's parameters, thereby reducing the risk of overfitting [22]. The model's performance is ultimately evaluated using the test dataset, which mirrors real-world, unseen data. This evaluation provides insights into the model's capacity to apply learned patterns to fresh data. In this study, the data is divided into three subsets using an 80-10-10 ratio, with 80% allocated for model training, 10% for parameter tuning and validation, and the remaining 10% reserved for final testing.

3.2 Model structure

Table 2. FFNN layer's structure

Layer (type)	Output shape
Flatten	(16, 120)
Dense	(16, 64)
Dense_1	(16, 32
Dense_2	(16, 24)

For the FFNN model structure shown in Table 2, since the input is 2D data, the Flatten layer is utilized to flatten it into 1D. The model then includes two fully connected (dense) layers, containing 64 and 32 neurons, respectively. Both layers use the "Rectified Linear Unit" (relu) activation function, which enhances the model's ability to learn complex patterns.

Table 3. GRU layer's structure

Layer (type)	Output shape
gru	(16, 24, 64)
gru_1	(16, 32)
Dropout_1	(16, 32)
Dense_1	(16, 24)

Table 4. LSTM layer's structure

Layer (type)	Output shape
lstm	(16, 24, 64)
lstm_1	(16, 32)
Dropout_1	(16, 32)
Dense_1	(16, 24)

Model GRU and LSTM are designed as in Table 3 and Table 4 with the input layer, and the other layer is configured to return the entire sequence of hidden values. To prevent overfitting, a Dropout layer with a rate of 0.2 is applied after each hidden layer. The model concludes with a Dense layer consisting of 12 neurons, designed to output 12 predicted values. In the case of the GRU model, the second hidden layer employs the relu activation function to improve the model's ability to learn and capture complex patterns. The model is optimized using the RMSprop algorithm, set with a learning rate of 0.001 and a gradient clipping threshold of 1.0.

On the other hand, the two bidirectional models, BiGRU and BiLSTM, share the same number and structure of hidden layers and dense layers compared to the unidirectional models, LSTM and GRU. The difference lies in the input layer in Table 5, where the Bidirectional models can learn bidirectional patterns. As a result, the number of neurons in the first layer is increased, yielding 128 neurons, which is twice the output of the first layer in the LSTM and GRU models.

Table 5. BiRNNs layer's structure

Layer (type)	Output shape
Bidirectional	(16, 24, 128)
Dropout_1	(16, 24, 128)
Bidirectional_1	(16, 64)
Dropout_2	(16, 64)
Dense_1	(16, 24)

3.3 Test results

After splitting the weather dataset, the performance results of the models are obtained through metrics such as MAE, MSE, and RMSE, as shown in Table 6.

Table 6. Comparison between forecasting models.

	MAE (kW)	MAPE (%)	RMSE (kW)
FFNN	50.769	6.903	75.960
GRU	42.781	6.427	59.931
LSTM	42.652	6.374	59.933
BiGRU	42.663	6.387	59.745
BiLSTM	43.264	6.455	60.891

Fig. 9 presents a visual representation of three distinct graphs, each depicting the comparative performance of five models across three different day types: weekends, holidays, and normal days. The results indicate that all examined RNN models demonstrate favorable outcomes with relatively low error levels. Particularly noteworthy is the impressive performance exhibited by the bidirectional RNN models, which excel in capturing temporal dependencies and producing accurate predictions on holidays and typical

working days. Conversely, the FFNN model, while characterized by its expedient training speed, exhibits a relatively higher degree of discrepancy due to its inherent limitations in effectively accommodating sequential data. Nonetheless, this disparity can be deemed acceptable, given the substantial magnitude of the dataset under consideration.

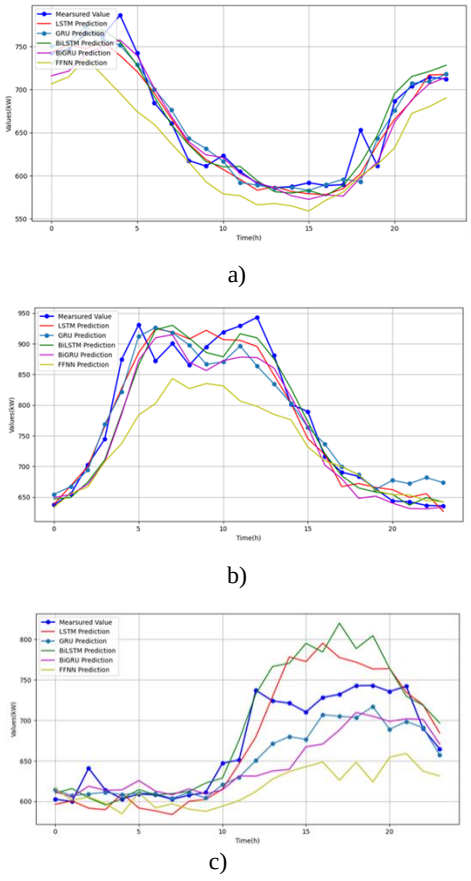


Fig. 9. Forecasting results of five models in three kind of days (holidays (a), normal day (b), weekend (c))

The findings underscore the efficacy of RNN models, particularly their bidirectional variants, in effectively capturing and leveraging temporal dependencies for accurate power consumption forecasting across diverse day types. In contrast, the FFNN model's suboptimal performance in handling sequential data yields comparatively higher prediction errors. Nevertheless, considering the ample scale of the dataset, this divergence can be deemed tolerable within acceptable bounds.

4. CONCLUSIONS

In our study, we delved deep into the field of deep learning for dynamic energy demand prediction in a residential building. Our exploration included a comparative analysis of five neural network architectures: FFNN, LSTM, GRU, BiLSTM, and BiGRU.

Through careful testing and evaluation, we discovered notable differences in the performance of these models. The RNN-based models emerged as the superior ones, outperforming the traditional FFNN model in the domain of time series prediction for energy demand. Their ability to understand and make use of patterns over time, while considering both past and future information, was crucial in achieving accurate predictions.

In contrast to FFNN and bidirectional models such as BiLSTM and BiGRU, LSTM and GRU models have yielded superior performance owing to their proficiency in capturing the sequential dynamics within energy demand data.

The enhanced performance of LSTM and GRU models can be ascribed to multiple factors. Primarily, their ability to learn long-term dependencies in the data is crucial for identifying patterns in energy demand. Second, they are less prone to overfitting, a common issue in energy demand forecasting. Third, they can handle the nonlinear relationships between variables, essential in energy demand forecasting, where the relationships between variables are often complex. In contrast, bidirectional models like BiLSTM and BiGRU, which consider both past and future information, have higher computational complexity and may not be suitable for the small feature set of the Tsukuba building dataset.

Furthermore, LSTM and GRU models are more interpretable and more accessible to train than bidirectional models, making them more suitable for practical applications. They are adaptable and straightforward to modify to fit different forecasting requirements and datasets.

These findings emphasize the importance of unidirectional models, such as LSTM and GRU, in capturing and leveraging the temporal dependencies present in energy demand data. Furthermore, our future research will build upon these insights to explore the interaction between energy forecasting and the economic dispatch of microgrids, thereby gaining a better understanding of energy management.

While this conclusion does not delve into the details of the microgrid electrical system, it provides a foundation for future research efforts. Refining and implementing unidirectional models like LSTM and GRU in real-world applications offers significant potential for improving energy demand forecasting and facilitating the integration of distributed generation to address changing energy needs.

In our forthcoming research endeavors, we intend to investigate the efficacy of hybrid models that synergistically combine the strengths of disparate neural network architectures. Specifically, we propose examining the performance of CNN-LSTM models, which capitalize on the spatial feature extraction capabilities of convolutional neural networks (CNNs) and the temporal modeling capabilities of LSTM networks. In addition, future work will focus on investigating attention-based models, such as the Transformer architecture, to further improve both the accuracy and interpretability of energy demand forecasting. By integrating these hybrid models with advanced

optimization techniques and real-time data analytics, we aim to create an energy management system that is more sustainable and effective and can adjust to the changing needs of contemporary power grids. Notably, the effectiveness of these models has been demonstrated in various time-series forecasting applications [23]-[26], where they have consistently outperformed traditional statistical models and single-architecture neural networks.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the Ministry of Education and Training for supporting their research through the 89 Project. They also extend their sincere gratitude to the Institute for Control Engineering and Automation at Hanoi University of Science and Technology for their continuous support of this research.

REFERENCES

[1] Dinh D.L. 2019. An Enhancement to Time Series Model for Short-Term Forecasting of Multi-Regional Electricity Load. GMSARN International Journal (13): 202-207.

[2] 2023. Operational situation in July 2023 and objectives and tasks in August 2023. VietNam: EVN.

[3] 2023. Northern region faces high risk of serious electricity shortage. VietNam: Viet Nam News.

[4] K. Vink, E. Ankyu, and M. Koyama. 2019. Multiyear microgrid data from a research building in Tsukuba, Japan. Scientific Data (6); 190020.

[5] C. Deina, JLF. dos Santos, L. H. Biuk, M. Lizot, A. Converti, H.V. Siqueira, and F. Trojan. 2023. Forecasting Electricity Demand by Neural Networks and Definition of Inputs by Multi-Criteria Analysis. Energies 16 (4); 1712.

[6] M. Abumohsen, A.Y. Owda, and M. Owda. 2023. Electrical Load Forecasting Using LSTM, GRU, and RNN Algorithms. Energies 16(5); 2283.

[7] L. Sehovac, C. Nesen and K. Grolinger. 2019. Forecasting Building Energy Consumption with Deep Learning: A Sequence to Sequence Approach. Electrical and Computer Engineering Publications; 166.

[8] S. Mahjoub, L. Chrifi-Alaoui, B. Marhic, and L. Delahoche. 2022. Predicting Energy Consumption Using LSTM, Multi-Layer GRU and Drop-GRU Neural Networks. Sensors 22(11): 4062.

[9] A. Nugaliyadde, U. Somaratne, and K. W. Wong. 2023. Predicting Electricity Consumption Using Deep Recurrent Neural Networks. ArXiv; 1909.08182.

[10] Bareth, R., Yadav, A., Gupta, S., and Pazoki, M. 2024. Daily average load demand forecasting using LSTM model based on historical load trends. IET Generation, Transmission & Distribution 18(5): 952-962.

[11] R. Bareth and A. Yadav. 2024. Day Ahead Load Demand Forecasting based on LSTM Machine Learning Model. Third International Conference on Power, Control and Computing Technologies (ICPC2T). Raipur, India, Jan 18.

[12] Hossein Abbasimehr, Mostafa Shabani, Mohsen Yousefi. 2020. An optimized model using LSTM network for demand forecasting. Computers & Industrial Engineering 143 (0360-8352): 106435.

[13] A.K. Shaikh, A. Nazir, and I. Khan, and A. S. Shah. 2022. Short term energy consumption forecasting using neural basis expansion analysis for interpretable time series. Sci Rep 12: 22562.

[14] W. S. McCulloch and W. A. Pitts. 1943. Logical Calculus of the Ideas Immanent in Nervous Activity. Bulletin of Mathematical Biophysics 5, 115-133.

[15] A. Geron. 2017. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. Sebastopol, California: Oreilly.

[16] S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. Neural Computation 9: 1735-1780.

[17] Duy L. B., Quang N. N., Van B. D., Tuan K. P. and Dinh D. L. 2024. Evaluating an Effectiveness of a Solar Power Plant Output Forecasting Model Based on LSTM Method Using Validation in Different Seasons of a Year in Vietnam. GMSARN International Journal 18: 114-122.

[18] K. H. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning Phrase Representations using RNN Encoder-Ecoder for Statistical Machine Translation. arXiv: 1406.1078.

[19] M. Schuster and K. K. Paliwal. 1997. Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing 45: 2673-2681.

[20] N. Sehrawat, S. Vashisht, and A. Singh. 2023. Solar irradiance forecasting models using machine learning techniques and digital twin: A case study with comparison. ScienceDirect 4 (2666-6030): 90-102.

[21] Japan Meteorological Agency, "Historical Weather Data Download," Available at: https://www.data.jma.go.jp/gmd/risk/obsdl/index.php (Accessed: 28 May 2025).Ewout W. Steyerberg. 2018. Validation in prediction research: the waste by data splitting. Journal of Clinical Epidemiology 103: 131-133.

[22] Lim, S.-C., Huh, J.-H., Hong, S.-H.; Park, C.-Y., Kim, J.-C. 2022. Solar Power Forecasting Using CNN-LSTM Hybrid Model. Energies 15: 8233.

[23] S. M. Mirjebreili, A. Solouki, H. Soltanalizadeh and M. Sabokrou. 2022. Multi-Task Transformer for Stock Market Trend Prediction. 12th International Conference on Computer and Knowledge Engineering (ICCKE). 17-18 November. Mashhad, Iran.

[24] Ailing Zeng, Muxi Chen, Lei Zhang, Qiang Xu. 2022. Are Transformers Effective for Time Series Forecasting? arXiv 2205: 13504.

[25] Ma, S., Zhang, T., Zhao, YB. et al. TCLN. 2023. A Transformer-based Conv-LSTM network for multivariate time series forecasting. Appl Intell 53: 28401–28417.